ETSI SmartM2M Technical Report 103717; Study for oneM2M; Discovery and Query specification development

Seung Myeong Jeong, Sunil Kumar, Andrea Cimmino, Raúl García Castro, Luigi Liquori, Marie-Agnès Peraldi-Frati, Enrico Scarrone, Joachim Koss, Flynn Bob

ETSI TR 103 717 V1.1.1 (2021-06)

SmartM2M;
Study for oneM2M;
Discovery and Query specification development

Reference
DTR/SmartM2M-103717

Keywords
INTEROPERABILITY, IoT, oneM2M, SAREF, Semantic

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

This Technical Report (TR) has been produced by ETSI Technical Committee Smart Machine-to-Machine communications (SmartM2M).

Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Introduction

oneM2M has currently native discovery capabilities that work properly only if the search is related to specific known sources of information (e.g. searching for the values of a known set of containers) or if the discovery is well scoped and designed (e.g. the lights in a house). When oneM2M is used to discover wide sets of data or unknown sets of data, the functionality is typically integrated by ad hoc applications that are expanding the oneM2M functionality. This means that this core function may be implemented with different flavours and this is not optimal for interworking and interoperability.

The objective of the present document [i.4] in conjunction with three other ones [i.1], [i.2] and [i.3] is the study and development of Semantic Discovery and Query capabilities for oneM2M and its contribution to the oneM2M standard.

The goal is to enable an easy and efficient discovery of information and a proper interworking with external sources/consumers of information (e.g. a distributed data base in a smart city or in a firm), or to directly search information in the oneM2M system for big data purposes.

1 Scope

1.1 Context for the present document

In order to enhance the semantic capabilities of the oneM2M architecture by providing solid contributions to the oneM2M standards, four Technical Reports have been developed. Each of them is the outcome of a special study phase.

The study and development of Semantic Discovery and Query capabilities for oneM2M and its contribution to the oneM2M standard is composed of four phases:

1) A **requirements** phase where requirements and use cases are formally identified and defined. As a minimum, this work includes discovery of specific information and of aggregated information, and interaction with external sources of data and queries. The oneM2M architecture [i.6], the oneM2M semantic approach [i.7], the current oneM2M capabilities and SAREF [i.8], [i.9], [i.10], [i.11], [i.12], [i.13], [i.14] are at the basis of these use cases and requirements. This work is documented in ETSI TR 103 714 [i.1].

2) A **study** phase where possible approaches (existing and new ones) to a discovery and data aggregation solution are analysed with respect to the use cases and requirements. In particular, the need to plug in the solution on the oneM2M standard drives the solution analysis, to determine the best approach to be followed. The present document also looks to the query and discovery mechanisms already available, starting from the ones defined by ETSI (e.g. the one included in NGSI-LD [i.15]) to extract (and potentially adapt) the applicable components and to assure a smooth interworking with non-oneM2M solutions. This is documented in ETSI TR 103 715 [i.2].

3) A **simulation** phase is conducted in parallel and "circular" feedback with respect to the study phase, with the goal to provide a proof of concept, run suitable scenarios provided by previous phases and a performance evaluation to support the selection/development of the Discovery and Query solution. The simulator/emulator and the simulation results are documented in ETSI TR 103 716 [i.3]. An extract of the simulation results is included ETSI TR 103 715 [i.2] and ETSI TR 103 717 [i.4] (the present document). A selection of the use cases includes a set of oneM2M relevant configurations scenarios to be considered for the simulation activity described below.

4) A **standardization** phase where the Discovery and Query solution is specified and documented in ETSI TR 103 717 [i.4] (present document).

The present document covers the fourth of the four phases and is related to the other documents listed below (the present document is highlighted in *italic* script in the list):

- ETSI TR 103 714: "SmartM2M; Study for oneM2M Discovery and Query use cases and requirements" [i.1];
- ETSI TR 103 715: "SmartM2M; Study for oneM2M Discovery and Query solutions analysis & selection" [i.2];
- ETSI TR 103 716: "SmartM2M; oneM2M Discovery and Query solution(s) simulation and performance evaluation" [i.3];
- *ETSI TR 103 717: "SmartM2M; Study for oneM2M Discovery and Query specification development" [*i.4*] (the present document).*

1.2 Scope of the present document

The present document develops the specification for the discovery solution selected in ETSI TR 103 715 [i.2] and which simulation is documented in ETSI TR 103 716 [i.3]. The present document specifies candidate solutions while the corresponding standardization proposals are contributed to oneM2M TS-0001 (Architecture) [i.5], oneM2M TS-0034 (Semantic support) [i.7], oneM2M TS-0033 (Interworking Framework) [i.18], oneM2M TS-0004 (Protocols) [i.19] (other oneM2M TS may be also impacted) with the help of the supporting companies active in oneM2M.

The present document is structured as follows:

- Clauses 1 to 3 set the scene and provide references as well as definitions of terms, symbols and abbreviations, which are used in the present document.

- Clause 4 describes the method used for developing the standardization proposal for oneM2M.
- Clause 5 presents an overview of the Advanced Semantic Discovery proposal, how to enhance the oneM2M architecture.
- Clause 6 illustrates oneM2M features with call flows among oneM2M entities with primitive exchange examples. This includes existing features from the oneM2M specs as well as the newly proposed features to completely explain the solutions.
- Clause 7 proposes new primitive, parameters, resource types and relevant procedures.
- Clause 8 provides some lessons learned and conclusions.

2 References

2.1 Normative references

Normative references are not applicable in the present document.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this Clause were valid at the time of publication ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI TR 103 714: "SmartM2M; Study for oneM2M; Discovery and Query use cases and requirements".

[i.2] ETSI TR 103 715: "SmartM2M; Study for oneM2M Discovery and Query solutions analysis & selection".

[i.3] ETSI TR 103 716: "SmartM2M; oneM2M Discovery and Query solution(s) simulation and performance evaluation".

[i.4] ETSI TR 103 717: "SmartM2M; Study for oneM2M Discovery and Query specification development".

[i.5] oneM2M TS-0001 (V4.8.0): "Functional Architecture".

NOTE: Available at http://member.onem2m.org/Application/documentapp/downloadLatestRevision/default.aspx?docID=31496.

[i.6] ETSI TS 118 101: "oneM2M; Functional Architecture (oneM2M TS-0001 version 3.9.0 Release 3)".

[i.7] oneM2M TS-0034 (V4.2.0): "Semantics Support".

NOTE: Available at http://member.onem2m.org/Application/documentapp/downloadLatestRevision/default.aspx?docID=31425.

[i.8] ETSI TS 103 264: "SmartM2M; Smart Applications; Reference Ontology and oneM2M Mapping".

[i.9] ETSI TS 103 410-1: "SmartM2M; Extension to SAREF; Part 1: Energy Domain".

[i.10] ETSI TS 103 410-2: "SmartM2M; Extension to SAREF; Part 2: Environment Domain".

[i.11] ETSI TS 103 410-3: "SmartM2M; Extension to SAREF; Part 3: Building Domain".

[i.12] ETSI TS 103 410-4: "SmartM2M Extension to SAREF Part 4: Smart Cities Domain".

[i.13] ETSI TS 103 410-5: "SmartM2M; Extension to SAREF Part 5: Industry and Manufacturing Domains".

[i.14] ETSI TS 103 410-6: "SmartM2M; Extension to SAREF; Part 6: Smart Agriculture and Food Chain Domain".

[i.15] ETSI GS CIM 009: "Context Information Management (CIM); NGSI-LD API".

[i.16] oneM2M TS-0033 (V3.0.0): "Interworking Framework".

NOTE: Available at http://member.onem2m.org/Application/documentapp/downloadLatestRevision/default.aspx?docID=29581.

[i.17] oneM2M TS-0004 (V4.3.0): "Service Layer Core Protocol".

NOTE: Available at http://member.onem2m.org/Application/documentapp/downloadLatestRevision/default.aspx?docID=32245.

[i.18] oneM2M TS-0003 (V4.2.0): "Security Solutions".

NOTE: Available at http://member.onem2m.org/Application/documentapp/downloadLatestRevision/default.aspx?docID=32192.

[i.19] oneM2M TR-0045 (V0.3.1): "Developer Guide: Implementing Semantics".

NOTE: Available at http://member.onem2m.org/Application/documentapp/downloadLatestRevision/default.aspx?docID=24354.

[i.20] W3C RDF 1.1 Primer

NOTE: Available at https://www.w3.org/TR/rdf11-primer/#section-triple

[i.21] W3C RDF Schema 1.1

NOTE: Available at https://www.w3.org/TR/rdf-schema/#ch_class

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the following terms apply:

Advanced Semantic Discovery Query (ASDQ): word in the Advanced Semantic Discovery Query Language (ASDQL) according to the Theory of Formal Languages.

Advanced Semantic Discovery Query Language (ASDQL): extension of the actual oneM2M Semantic Discovery Query Language (SDQL), which has to be suitable enough to describe queries that will be resolved in a *cooperative* way by a distributed network of CSEs

NOTE: Each CSE involved in the resolution participates in resolving subqueries and aggregating results by coordinating and cooperating among each other's.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

AE	Application Entity
API	Application Program Interface
ASD	Advanced Semantic Discovery
CSE	Common Services Entity
ETSI	European Telecommunications Standards Institute
HTTP	Hypertext Transfer Protocol
IN	Infrastructure Node
IN-CSE	Infrastructure Node - Common Services Entity
IoT	Internet of Things
P2P	Peer-to-Peer
RDF	Resource Description Framework
SDA	Semantic Discovery Agreement
SPARQL	Simple Protocol and RDF Query Language
SR	Semantic Recommendation System
SRT	Semantic Routing Table
TR	Technical Report
URI	Uniform Resource Identifier

4 Method for Specification Development

The study and development of Advanced Semantic Discovery and Query capabilities for oneM2M and its contribution to the oneM2M standard is composed of four phases, which are described in clause 1.1 (requirements phase, study phase, simulation phase and standardization phase). The results of each phase are documented in Technical Reports [i.1], [i.2], [i.3] and in the present document [i.4]. These 4 phases with their TRs form a kind of chain, where one phase is built upon the former one and the present document TR 103 717 [i.4] includes the final study objective as it develops the specification for the discovery solution selected in ETSI TR 103 715 [i.2] and which simulation is documented in ETSI TR 103 716 [i.3]. The present document provides candidate solutions while the corresponding standardization proposals are contributed to oneM2M TS-0001 (Architecture) [i.5], oneM2M TS-0034 (Semantic support) [i.7], oneM2M TS-0033 (Interworking Framework) [i.18], oneM2M TS-0004 (Protocols) [i.19] (other oneM2M TS may be also impacted).

In order to check the consistency between the 4 TRs, a working tool (2 Working Documents) has been used. This tool acts as a checklist through mappings between the relevant clauses of the TRs. It does not force the mapping of all requirements or solution approaches from one TR into another TR but serves as a **checklist** in order not to forget important ones on the way to the final reporting of standardization proposals to enhance the semantic capabilities of the oneM2M architecture (see figures 4-1, 4-2, 4-3 and 4.4).

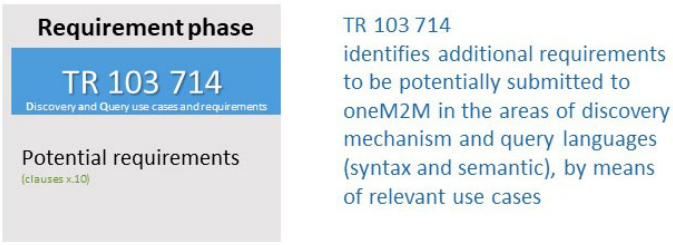

Figure 4-1: Requirement phase

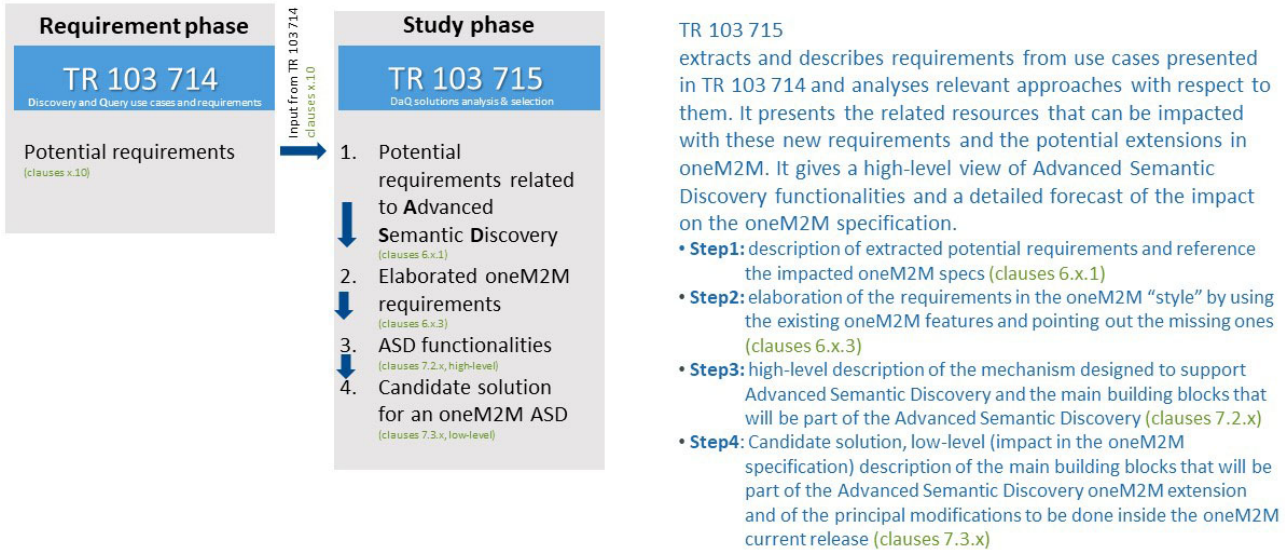

Figure 4-2: Study phase

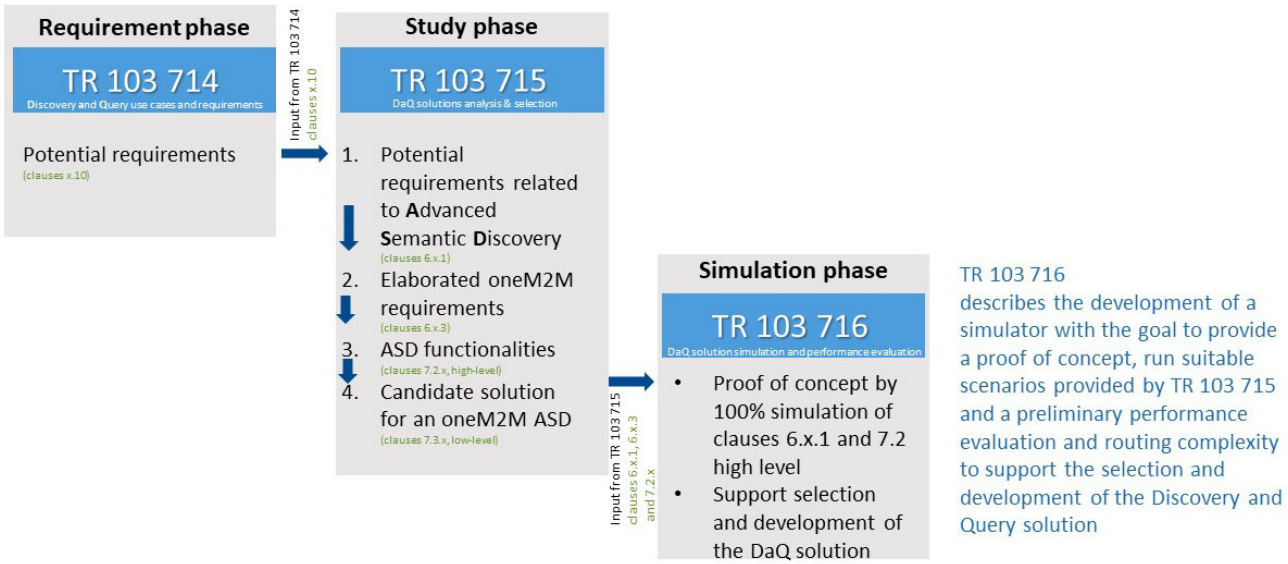

Figure 4-3: Simulation phase

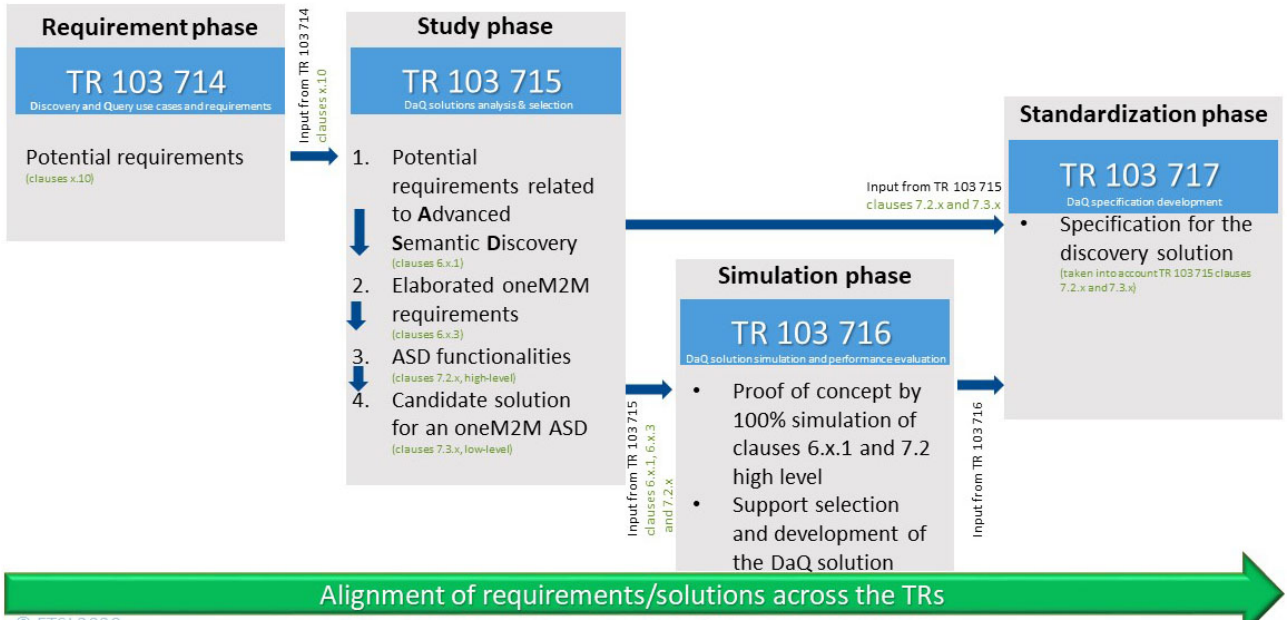

Figure 4-4: Standardization phase

Presentations to and discussions with relevant oneM2M Working Groups about the proposed ASD enhancements also build an important part of the working method for the development of standardization proposals.

5 Overview of the Advanced Semantic Discovery

5.1 Introduction

This clause explains why the Advanced Semantic Discovery (ASD) is needed, and how the ASD can be beneficial to oneM2M systems. To start explanation, the concept of ASD needs to be defined first for common understanding. As the name stands for, this is a sort of extension to the semantic discovery that oneM2M already supports [i.7]. The idea of advanced or extended here comes from the shortcomings from the existing semantic discovery mechanisms.

The targeted data or information for semantic discovery by the specification is RDF triples and their stored in <*semanticDescriptor*> resources in distributed CSEs. For <semanticDescriptor> resources in a CSE, if no specific resource identifier for the discovery is given, at least the resource root (i.e. <CSEBase> resource) of the CSE can be set for the discovery target (i.e. ***To*** parameter). However, when the Originator wants to find information with the semantic discovery but has no idea which CSEs possibly have matches, it gets really difficult to send the request.

Therefore the essence of the ASD is that the ASD provides capabilities to request semantic discovery to whom would know the candidate targets to perform the semantic discovery. The CSE who gets the ASD from the Originator selects CSE(s) to execute the query and send back the aggregated, if needed, results. When the ASD carries complex query which requires different CSEs to answer partially, the original query can be split into pieces and sent to those CSEs.

Note that semantic discovery in this document also covers the also covers the semantic query mechanism in [i.7].

The other terminology clarifications might also be needed for the present document, especially the terms that have been used in the other TRs such as [i.2]. Firstly, routing in this document, which specifies oneM2M solutions as candidate refers to finding CSEs that can execute ASD queries. This is different context from the traditional IP routing. In the network routing, data packets get routed on each router to another router toward the target IP address. On the contrary, as the concept depicted above, in case of the ASD, the Originator cannot point out the targeted CSEs.

Also, fan-out in this document implies the fan-out concept in existing oneM2M specifications. The group management feature in oneM2M uses fan-out concept. There can be a group of oneM2M resources and applications can access to those member resources with one request. For example, there are thousands of sensors deployed in a field. Once those are grouped as a <group> resource, then an application gets retrieve the latest sensor readings via a <fanOutPoint> child resource of the <group> resource. When the Hosting CSE of the <group> resource gets the request, it fans-out the request to all the member resources. Fan-out here means to send the same request to different members. Of course, in the primitive level, since ***To*** parameter settings in those request primitives are different, but ***Operation*** and ***Content*** parameter values are the same. This is one of the cases for the ASD. Also, sending different queries, that are fragmented from the original one, is also possible. This is called fan-out in this document, too

5.2 Scenario for Advanced Semantic Discovery

5.2.1 Overall scenario

This clause re-illustrates the use case scenarios that are defined in [i.2] to be more oneM2M standard compliant. Hence the clause 6 proposes the oneM2M solutions to fulfil the scenarios in oneM2M architecture.

The overall scenarios consist of the followings:

- Entity registration and other resource creation
- Semantic routing table updates and propagation
- Advanced Semantic Discovery handling involving query fragmentation

The figure 5.2.1-1 describes the entity registration and the semantic routing table management procedures. It is assumed that, for simplicity and as illustrated from the original scenario in [i.2], the discovery target type is the AE. When AEs register to its Registrar CSEs a new routing table record gets created on the Registrar. The record can contain any information regarding the AE. For example, the App-ID, which is different from AE-ID, and labels can be added. If the AE represents a sensor application, some meta information on sensor measurement could also be contained.

Once, the new routing information is created, depending on the agreements among CSEs, the new routing information gets propagated to other CSEs. This is performed by CSEs per AE registration by pre-configurations. As part of the

agreement between the AE and the CSE, the AE lets CSE know which resources should be incorporated in the routing table and which CSEs can have and use the information.

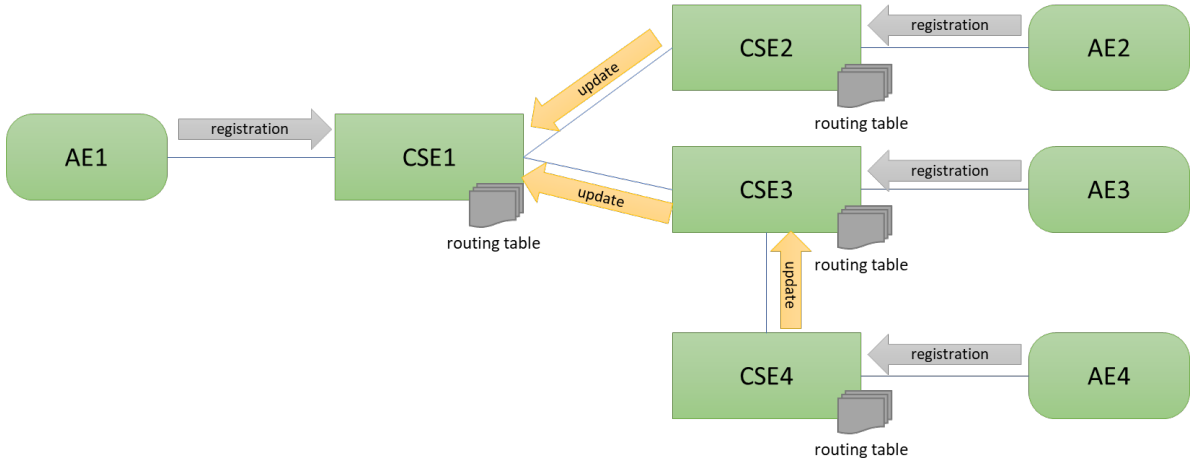

Figure 5.2.1-1: Routing table updates for Advanced Semantic Discovery

Before requesting the Advanced Semantic Discovery, an Originator selects the CSE to handle the request. In the ordinary oneM2M discovery, the Originator knows the targeted CSE which host interested resources. However, in the Advanced Semantic Discovery, the Originator does not need to know the CSEs since the request would be handled by multiple CSEs using the Semantic Routing Tables. Therefore, the Originator may send the discovery request to its Registrar CSE by default, unless it is aware of the identifiers of the other CSEs who may be better having bigger SRTs.

When the Originator requests for an Advanced Semantic Discovery, that can be recognized by corresponding parameters in the request, the CSE resolves the discovery query. If the discovery is completely resolved by the CSE, it sends the response back to the Originator. Otherwise, the CSE fans-out the ASD request, optionally involving query fragmentation, so it can be handled by other CSEs by looking up its routing table and the query in the ASD request. The fan-out ASD requests are sent to those CSEs as new oneM2M requests, so the CSE (e.g. CSE1 in the figure 5.2.1-2) waits and aggregates the responses to send the final response to the Originator. The fragmented query can also be split into smaller queries, however constraints, such as time limit to send back the response to the Originator, would be kept in the newly initiated requests.

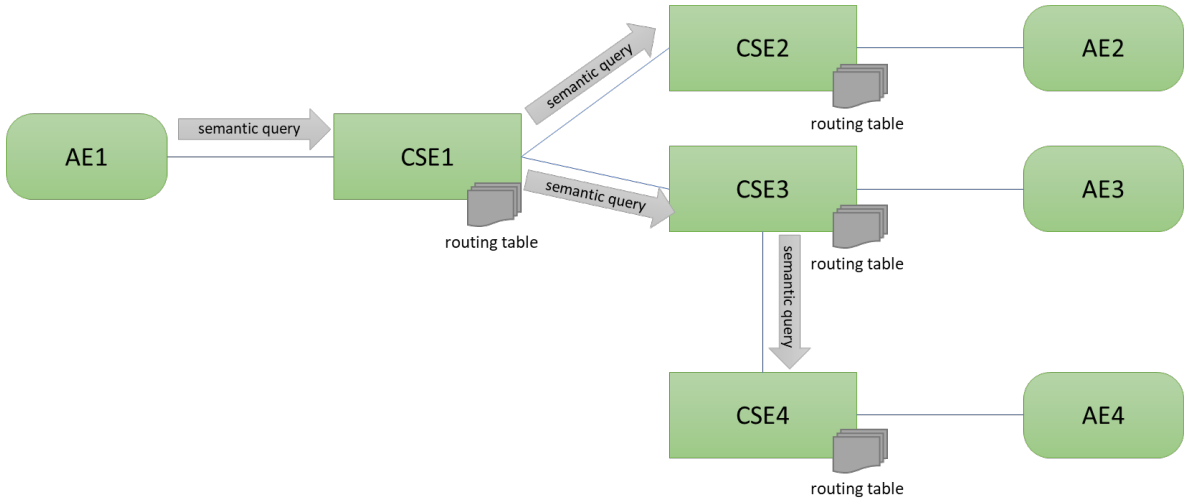

Figure 5.2.1-2: Advanced semantic query execution over multiple CSEs

5.2.2 ASD targets

An Advanced Semantic Discovery target type can be the AE, which is represented as an <AE> resource in oneM2M platform (i.e. CSE). An <AE> resource contains the information on the AE which can be device applications or service

applications. As the oneM2M architecture [i.5] defines, an AE can have information for its hosted device (e.g. child <node> resources) and its application data (e.g. <container> and <contentInstance> resources) as well as its context information (e.g. pointOfAccess). Presumably, child or descendant resources of the <AE> resource can be used for the advanced semantic discovery.

Likewise, oneM2M CSEs can be discovered since they are represented as <remoteCSE> resources and could have similar child resources like an <AE> resource. When there is a request seeking for a CSE having particular information, the same advanced semantic discovery can be realised as it is possible for the AEs.

These would the typical implementation cases when <AE> and <remoteCSE> resources represent a device or a service and the ASD searches for them. However, technically SPARQL query executions on semantic data can retrieve any information as intended, target can be anything described in the semantic data.

5.2.3 ASD scope

As mentioned briefly in the clause 5.2.1, existing oneM2M discovery can be performed for a single CSE. In Advanced Semantic Discovery, the scope of CSEs, which can be more than one, to execute the query can be indicated in the request. It can be the number of CSEs or the hops of CSEs in the entity hierarchy. This needs to be limited to avoid processing overheads by the CSEs. If the scope is not suggested by the Originator, the system default should be applied.

5.2.4 ASD result type

In oneM2M, the semantic discovery and the query are supported [i.7] which have limited search scope as one targeted CSE in general. Both of them execute the SPARQL query that is carried in oneM2M request primitive, however the results are different. The semantic discovery, like ordinary oneM2M discovery does, returns identifiers of the matching oneM2M resources. In this case, identifiers of the <*semanticDescriptor*> resource parents are provided in the response. In case of the semantic query, the SPARQL result itself is returned in the oneM2M response content.

This could be also applied for advanced semantic discovery. Depending on the indication from the Originator, the platform can return the matching resource identifiers or information from the SPARQL query execution.

5.3 Necessary technologies for implementation

5.3.1 Ontology for Advanced Semantic Discovery

The Advanced Semantic Discovery, considers the distributed environment where CSEs have different semantic information sharing through the oneM2M overlay network topology. The system does not restrict to particular ontology, as different domains requires different ontologies and all CSEs are not required to interpret all the ontologies. However, in some cases a CSE is expected to access the newly encountered ontology as described in Clause 6.2.2.

5.3.2 SPARQL query fragmentation and result aggregation

The ASD considers the environment, where the semantic data is distributed across the CSEs and the organisation and management of semantic resources vary based on each CSE. In this scenario, there can be cases of retrieving semantic data representing a single oneM2M resource, might be reorganized and distributed among different CSEs and vice versa. In addition, the acquisition of semantic data via SPARQL becomes complex, considering different parameters that can be defined based on SPARQL 1.1 features, which are different from oneM2M resource constraints.

The SPARQL query fragmentation is an advanced feature of ASD which is performed when the ASD query is complex. It involves acquisition of required semantic and oneM2M resource data in a distributed manner. If the complex query involves the response to be acquired from multiple CSEs, having either same or different ontologies, and aggregated on return, then the query can be fragmented upon request, into smaller and simplified subsequent queries, targeting more restricted number of CSEs than before, in order to acquire the result. It considers the Hosting CSE to be able to parse the SPARQL query and extract the meta information required for SRT lookup and then selecting the target CSEs for each subsequent query. Here the SPARQL query and the SRT plays a vital role in determining the CSEs which executes the subsequent query. Upon the execution of each subsequent query, the responses of corresponding target CSEs need to be aggregated in order to provide the result of the original unfragmented query. The subsequent queries are

considered as a part of ASD process which do not communicate with AE originator during their execution. Therefore, the aggregated result needs to be translated as the response to the original ASD request sent by the Originator AE.

6 oneM2M feature descriptions

6.1 Semantic routing table for ASD

6.1.1 Overview

The SRT is used by a Hosting CSE, that hosts a resource to handle an ASD request, to find CSEs that execute queries in the ASD. Since the ASDs include SPARQL queries for the distributed RDF triples [i.22] over multiple CSEs, SRTs are maintained by the data from RDF triples from pertaining oneM2M resource instances (i.e. <semanticDescriptor> resource). This sub-clause describes how it looks like and how a SRT gets populated among CSEs.

6.1.2 Data sources for SRT

As mentioned, the targeted data for the advanced semantic discovery is semantic data (i.e. RDF triples) that are contained in <*semanticDescriptor*> resources in different CSEs. Therefore the final goal of an ASD from the Originator is to find CSEs that would have ASD matching semantic resources and get results from them, and it can be done by a single CSEs not those distributed CSEs.

By the specification, a <*semanticDescriptor*> resource can be created as a child resource of different parent resources (e.g. AE, container). This parent-child resource relationship explains that a <semanticDescriptor> resource describes or annotates the parent resource. From the example resource tree in Figure 6.1.2-1, if an ASD got a match for the <semanticDescriptor> resource of a container, then the data the Originator is looking for is contained in the container.

It is worthwhile to note that in oneM2M system, each resource including the semantic resource has ownership. This is because not all semantic resources get propagated in SRTs. There needs policies to define which semantic resources gets propagated to whom by which condition or event. Those policies can be given by the owners of the resources. There are several concepts that define the ownership such as holder (owner), creator and resource hierarchy. This topic is elaborated in Clause 6.1.4.

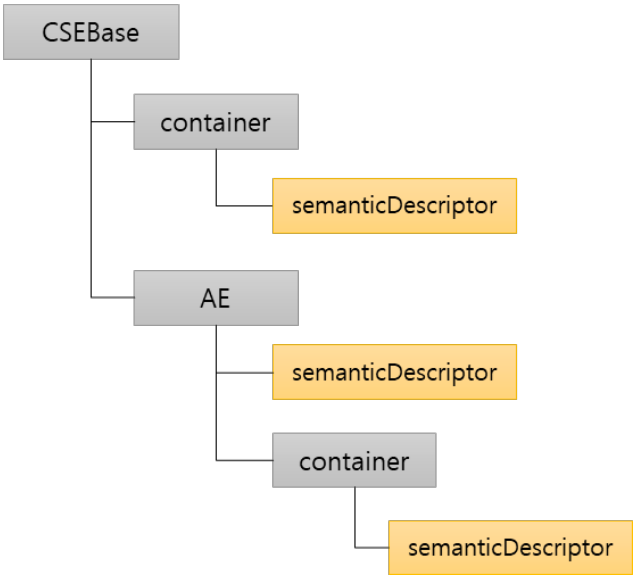

Figure 6.1.2-1: A resource tree with semanticDescriptor resources

6.1.3 Elements of routing records

6.1.3.1 Mandatory elements

6.1.3.1.1 CSE identifier

Since the major objective to have SRTs is to select CSE(s) to perform the semantic queries in ASDs, there is the CSE-ID along with the RDF resource summary. When there is a match with the summary, the associated CSE-ID is used to send subsequent ASD requests. The ***To*** parameter values of subsequent ASDs consist of CSE-ID and ASD request handling resource identifier. Further details are described in Clause 6.2.4.

6.1.3.1.2 RDF resource summary

Here the summary can be defined as the concise information, about the triple data, stored in a particular CSE, which determines the possibility of a valid (not-null) response retrieval based on a specific SPARQL query. Different levels of granularity can be involved in the summary, depending on the ASD criteria and fragmentation. The goal of this summary is to identify the candidate CSEs that can return the response for a given SPARQL query. This will provide a non exhaustive list of CSEs, which lead to faster ASD execution, restricting from redundant query fanouts. However, the summary should be able to accurately represent the actual triple data, stored in the CSEs. There can be different approaches for defining a summary, which can vary based on ASD policies.

One approach of defining summary is to involve the type definitions of the triple data stored. These type definitions can include the definitions from the ontologies, which the particular CSE is utilizing to define the triple data or any other schema involved in representing the triples. This will organize the information based on the types or ontologies which the target CSEs utilize and the hosting CSE will be able perform selection using this information. Using this approach the entries of SRT, in worst possible scenario, can be as big as the size of the ontologies or the schema using by the target CSEs.

Second approach involves including some selected part of the triple data in the SRT. This can involve any information which may be considered as focal point of the triple data to be queried. In this case the hosting CSE will create an anticipation or generalization about the triple data stored in the target CSE and will perform actions accordingly. The advantage of this approach is that the entries stored in the SRT can be made concise by selecting minimal information. However, the process of anticipation or generalization can become a challenge when the data is more diverse across the CSEs, having triples annotated in complex structures and are distributed among many CSEs.

Last approach involves definition of meta information which provides the mapping with the actual triples defined in the CSE. This mapping should be unanimously identified by all the CSEs in the network so that any hosting CSE can interpret the entries in the SRT and perform selection accordingly. The mapping can involve the URIs, keywords or any other identifiers, which can accurately represent a valid semantic resource in the database. This approach provides a lot of flexibility in defining the summary, however the mapping should support all the groups (identified by ontologies), and therefore, the triple data stored in the CSE in the considered network. Hence explicit mapping is required with each group of triple data, which can be a challenge as the number of groups (ontologies) increases. In addition, the mapping needs to be propagated among all the CSEs for unanimous interpretation of summaries in the SRT.

For this usecase, the first option is considered to represent a summary in the SRT. This is because the considered environment is distributed, having different ontologies. Also, the size of the SRT is compromised as to support wide range of diversity in the triple data. Table 6.1.2.1.1-1 shows the example summary of the triple data involving three different CSE as defined in the column CSE-ID. In the column ‘RDF Resource Summary’, each entry includes the list of URIs, representing RDF Class or a RDF Property [i.23]. Using this structure, two types of information can be extracted. First one is the URIs of the ontologies, identified as the prefix of the URI, used by CSE to represent its triple data. Second is the types of the RDF resources, currently involved to represent the RDF triple data. In this example total three prefixes are used which are ‘http://www.xyz.org/ontology/base-v1’, ‘http://www.abc.org/ontology/city-v1’ and ‘http://www.onem2m.org/ontology/base-v1’, representing three different ontologies. The postfix (value after ‘#’ in the URI) determines the type or RDF resource. The general convention of identifying the type is such that the postfix with a capitalized initial letter indicates that the URI represents an RDF Class and an RDF Property otherwise. In the example the postfix ‘Device’, ‘Service’ and ‘Function’ represents a RDF Class URI and postfix ‘location’ and ‘manufacturer’, represents the RDF Property URI.

Table 6.1.3.1.2-1: Example summary in the SRT

CSE-ID	RDF Resource Summary

CSE2	http://www.xyz.org/ontology/base-v1#Device http://www.xyz.org/ontology/base-v1#Service http://www.xyz.org/ontology/base-v1#location
CSE3	http://www.abc.org/ontology/city-v1#Device http://www.abc.org/ontology/city-v1#Function http://www.abc.org/ontology/city-v1#manufacturer
CSE4	http://www.onem2m.org/ontology/base-v1#Device http://www.xyz.org/ontology/base-v1#location

6.1.3.2 Optional elements

The cardinality per each type in the summary can be kept in the Semantic Routing Table. When the cardinality information presents, the SRT hosting CSE should take into account the cardinality when it selects CSE(s) for semantic routing. Bigger cardinality means there would be more matching semantic data, so the CSE would prefer the associated CSE over others.

In the similar sense, the number of hops between the SRT hosting CSE and the remote CSE can be referred to select CSEs for ASD fan-out. When a CSE is in closer than others in terms of hops, the ASD response could be returned quicker than others generally.

Different relationship types of Semantic Discovery Agreement can also be stored and used by the CSE. For example, when fanning-out an ASD to the Provider CSE (e.g. Customer-Provider relationship), there could be more information to find better ASD handling CSEs as candidates. The types with the routing examples are illustrated in TR 103 715 [i.2] and TR 103 716 [i.3].

Including the optional elements of the SRT, the table in this present document aligned with the original tables in TR 103 715 [i.2] and TR 103 716 [i.3] from the information point of view.

6.1.4 SRT update

A SRT needs to be updated so a SRT can find proper target CSEs that execute ASDs. When there is newly available RDF resource summary on a CSE, it can be propagated to other CSEs so those CSEs have up-to-date information to distribute ASDs.

In general, a SRT can be maintained by two methods which is depicted in the Figure 6.1.4-1:

- Explicit update by a CSE that hosts <semanticDescriptor> resources
- Subscription set by a SRT hosting CSE and notification by a <semanticDescriptor> resource hosting CSE

The CSE4 who has something to advertise into the other CSE's SRT can explicitly sends an Update request to the CSE1 so the CSE1 updates its SRT.

Note that the SRT in the diagram is a temporal virtual resource name for ASD routing table maintenance. In oneM2M specifications, virtual resources represent a specific feature without pertaining resource representation. Therefore, it is handy to define a set of procedures without defining a REST resource especially when there could be different implementations. One benefit to exploit a virtual resource concept here is that list update gets easier. When there is a long list of data in a resource attribute, to update such as adding or deleting item(s) from the original list, the Update request need to contain the entire items having new and old items. With the virtual resource, the Originator can include an item to add or delete with the indication whether it is adding an item or deleting it.

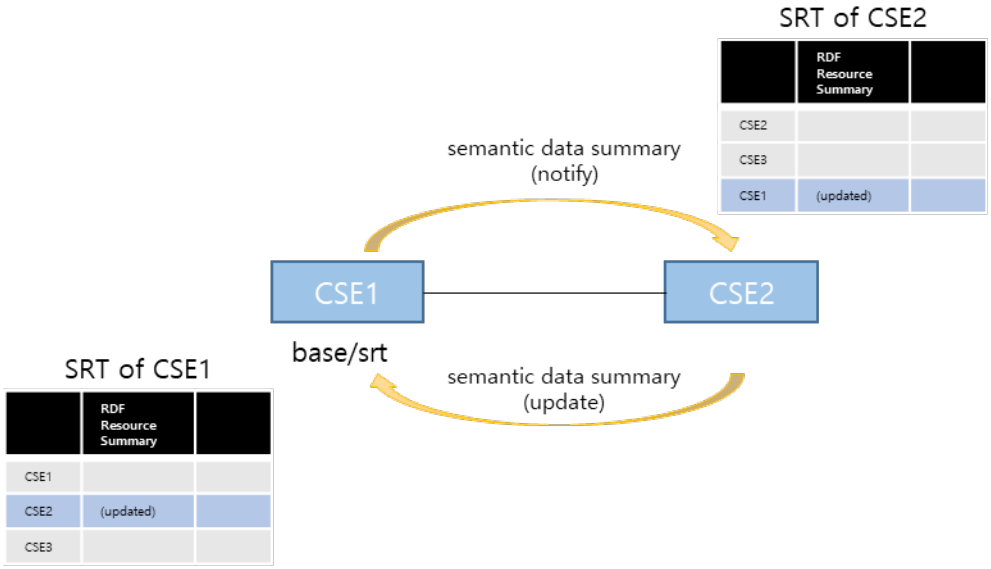

Figure 6.1.4-1: SRT update with virtual resource

Alternatively, subscription/notification feature can be used. It is used by the SRT Hosting CSEs, not the semantic data Hosting CSEs. Therefore the two different options can be used in different use cases – who initiates the SRT updates. In the Figure 6.1.4-1, the CSE1 subscribes an event to keep updating its own SRT regarding CSE3's semantic data by creating a <subscription> resource with corresponding notification event criteria. By the existing subscription/notification mechanism in oneM2M systems, there is a single resource (i.e. subscribed-to resource) for event subscription. In this case, the <*SRT*> virtual resource is the subscribed-to resource so a <subscription> resource creation request is configured to have the ***To*** parameter as the resource identifier such as "CSE1/base/SRT".

Another benefit to have SRT as a virtual resource is that CSE1 knows the request target since virtual resource names are given in the specification. On the other hand, if the SRT is represented in a normal resource, the resource name is given by the entity who creates that so there is the discovery first to get access to the SRT. Also in case of normal resource, there would be more than one resource instances.

oneM2M provides different event types for notifications, but to trigger notifications for new semantic data summary there needs a new event type definition to the notificationEventType of a subscription resource. The new event type "Update to semantic data summary". Basically new summary data can be notified per event. There could be filtering per semantic discovery agreement between (e.g. CSE1 and CSE3).

Whatever the methods, there is no limitation for CSEs involving the propagation. It could be done between Registrar and Registry in zero hop. Also, could happen between remote CSEs (e.g. CSE1 and CSE4 in 3 hops). Note that the term and definition hop is defined in oneM2M TS-0001.

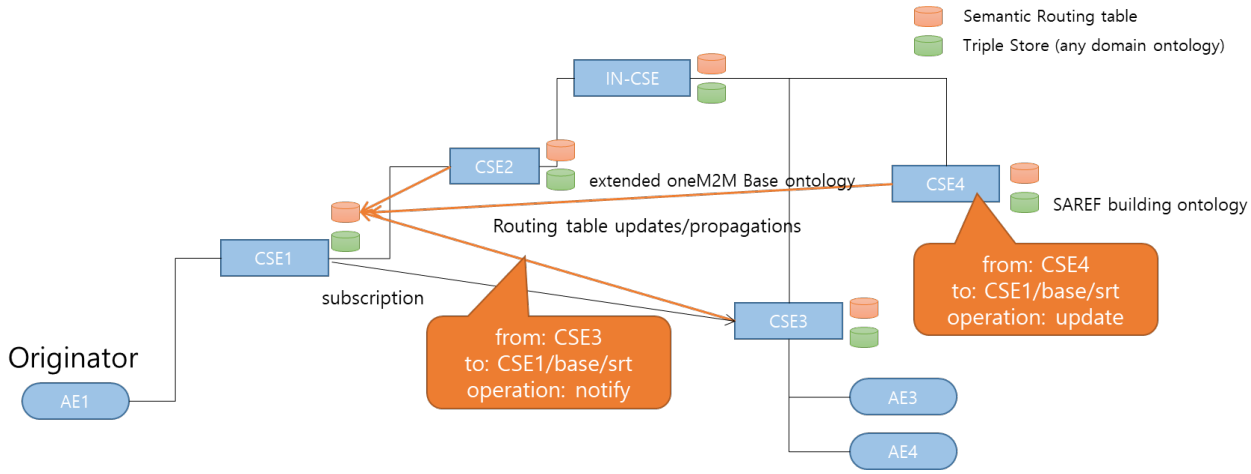

Figure 6.1.4-2: SRT update examples

Once a new semantic data summary gets noticed by a CSE, then it can be propagated by an update or notification. Noticing new semantic summary involves a couple of aspects. Firstly, there are more than one <*semanticDescriptor*> resources. Different resources could have duplicated summary (e.g. RDF Class URI), so each manipulation on a <*semanticDescriptor*> resource needs to be monitored by the CSE. For the duplicated summary, from the Hosting CSE perspective, it is not required to propagate that data.

Secondly, different AEs create their own <semanticDescriptor> resources and they would have different policies on SRT updates. This means not all RDF triples in <semanticDescriptor> resources get summarized and sends to other CSEs. Based on the AE's policies that are set a priori on a CSE (e.g. new *semanticDataPropagationPolicy* attribute of a <AE> resource), summarization methods (Clause 6.1.3.1.2) could be chosen and specific group of CSEs can get summary updates involving semantic discovery agreement (e.g. P2P CSEs).

6.2 ASD Handling

6.2.1 Overview

The core element of an ASD request is the embedded SPARQL query, which aims at retrieving the requested triple data from the considered CSEs. The SPARQL query could be executed on the selected CSEs if following considerations are fulfilled:

- The SPARQL query should contain proper definitions of prefixes involved.
- The SPARQL query should contain at least one Class type (RDF Class) or Property type (RDF Property) as a condition to be matched in the WHERE Clause.

The first consideration is focused on retrieving the list of ontologies involved in the query. Based on these prefixes, and routing table, the hosting CSE will be able to select the target CSE, where the query will be sent for execution. The second consideration is related to the case when the ASD request specifies RDF Summaries to be matched in the triple data. As described in Clause 6.1.3.1.2, the entries in the routing table will include the Class or Property URI if their corresponding instances are populated in the database. Therefore the URI of the RDF Class or Property type in the query will be used to select the target CSEs which contain the respective triple data. In addition, it is also an important factor in SPARQL query fragmentation, which is described in Clause 6.2.3.

6.2.2 ASD request handling with criteria matching

The ASD can be performed with different matching criteria in the SRT, as each criterion has different impact such as the search complexity, query fan-out, response time etc. Three matching criteria can be possible to select the target CSEs. These criteria can be selected by the Originator or by the local policies of CSEs. These criteria can be considered as a selection to have a compromise, mainly between the lookup complexity and the query fan-out.

The first matching criteria is to select the target CSEs based on the Ontology URI. All the prefixes from the query are extracted and searched in the SRT. Then those CSEs are selected which contain all the URIs extracted from the query. In this case, more CSEs will be selected than by using the other matching criteria, as the CSE are selected only based on the ontologies, of which the instances are available. There is an important point to note that the CSE will not be selected if it does not contain any instance of required ontology, even if the CSE contains, can access or can interpret that ontology. This emphasizes the preference of querying the instances as triple data in the ASD than the ontologies, as ontologies are assumed to be accessible on the Web.

The second criteria involves matching the summary of the triple data available in the CSE. In this case, the summary involves the URI of a (RDF) Class type or Property type, of which at least one instance is available in the database. The URIs of all the Class and Property types are extracted from the query and matched with the summary given in SRT. Those CSEs, will be selected of which, the summary matches with the Class and Property types, extracted from the query. Here note that the summary does not ensure that the valid triple data will be returned in response. Rather it filters out the irrelevant CSEs based on the query and selects those which matches with the query parameters. Nonetheless, there can be case when the CSE is selected based on the summary and it returns null value upon the query execution.

The third criteria involves matching summary with semantic inferencing features, which allows a hosting CSE to select the target CSE by performing inferencing techniques on the ontologies involved in the query. This requires that hosting CSE is able to access the ontology and perform inferencing. The target of inferencing is to extract sufficient information to select a target CSE. Note that, this criterion does not requires or considers the ability of target CSE to be able to

perform inferencing. This process is limited to the selection of target CSEs and the query execution and response are dependent on the target CSEs' ability to interpret the query and perform inferencing.

Figure 6.2.2-1(a) shows the example of ASD query with request of matching ontologies. Here the ontology URI is found in both entries of CSE2 and CSE4 so the ASD is sent to both of the CSEs. Whereas in Figure 6.2.2-1(b), the ASD query involves matching summary, in which the matching Class and property types exist only in CSE2. Hence in this case the ASD request is sent to CSE2 only. This emphasizes the above defined considerations for SPARQL query.

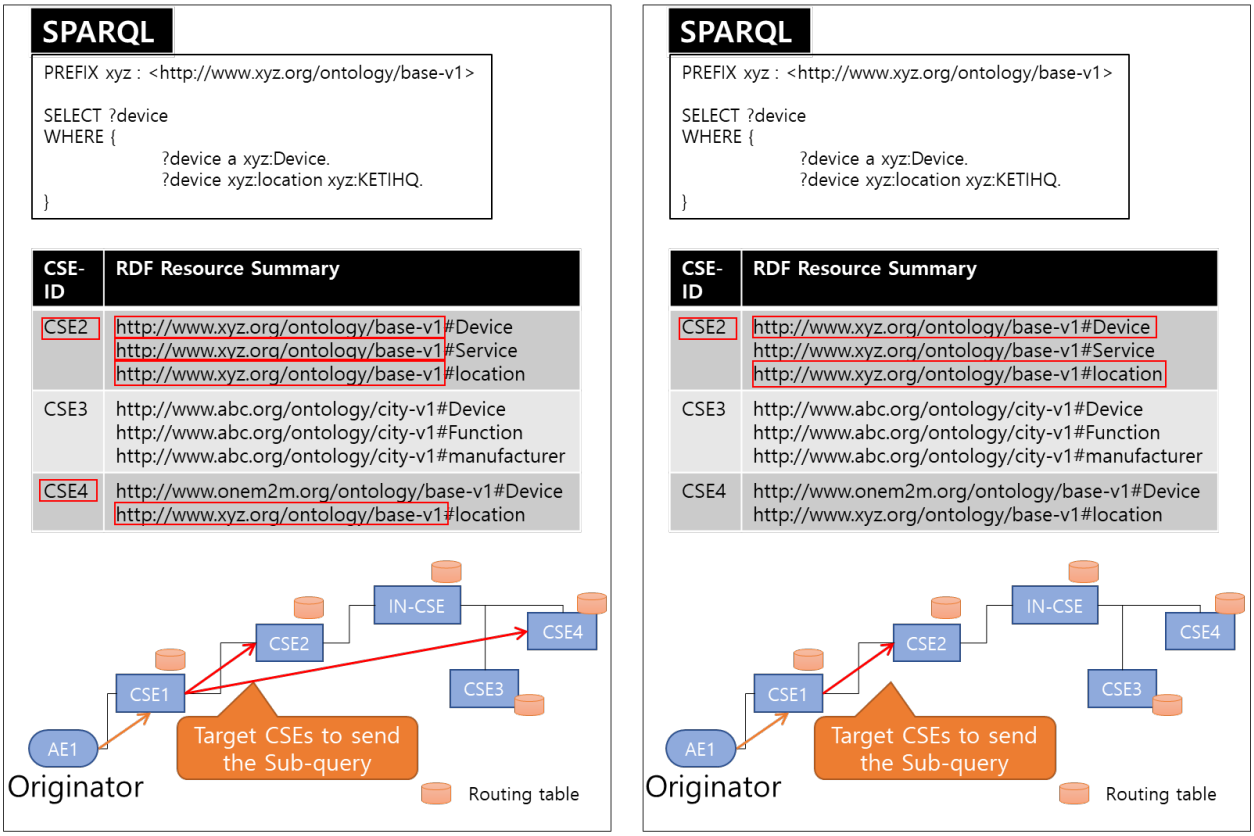

Figure 6.2.2-1: ASD handling based on matching ontologies and matching summary

Figure 6.2.2-2 shows the example of ASD query where the hosting CSE performs inferencing on the Class hierarchy of an ontology in order to select the target CSE. The SPARQL query involves a statement in the WHERE clause, specifying a type 'xyz:Device' (which after dereferencing using the prefix given in the query, becomes 'http://www.xyz.org/ontology/base-v1#Device'). Even though it can be seen in the SRT that CSE2 contains URIs, from the same ontology, as specified in the SPARQL query, no such RDF Class or Property URI can be completely matched with 'xyz:Device'. This is the case where inferencing techniques are required, in order to inquire, at a higher granularity level, whether CSE2 is a suitable candidate for executing the given SPARQL query. In this case, using the prefix URI (http://www.xyz.org/ontology/base-v1), the hosting CSE1 accesses the xyz ontology to perform inference in order to interpret the class hierarchies involving Classes 'xyz:Device' and 'xyz:SmartDevice'. After inferencing the ontology, it can be realized that 'xyz:SmartDevice' is a sub-class (rdfs:subClassOf [i.22]) of 'xyz:Device', which can be matched with the first entry of CSE2 of the SRT (as by using the Property rdfs:subClassOf, the class hierarchy in an ontology defines taxonomy of same concept, classifying generic to specific concepts, when the hierarchy is traversed from super Class to sub Class). Based on this interpretation CSE2 can be selected as a target CSE for query execution.

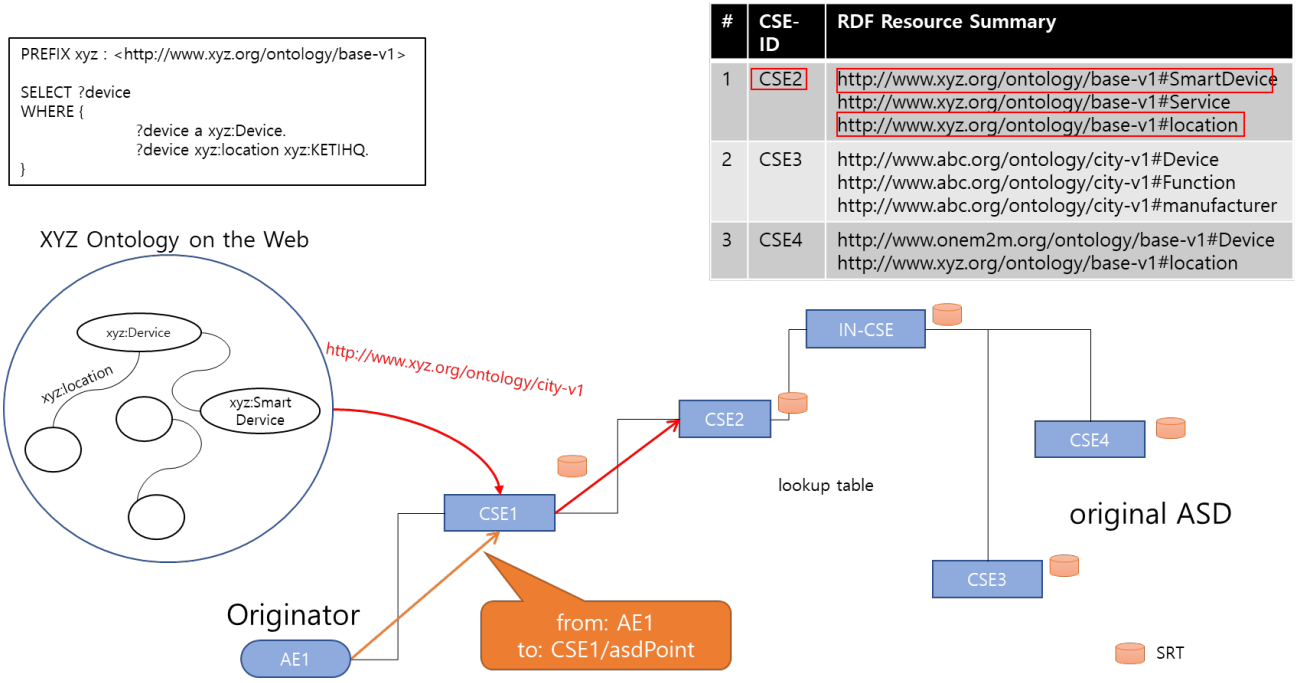

Figure 6.2.2-2: ASD handling based on matching summary with inferencing

6.2.3 ASD Fragmentation

In order to control the request fanout, hosting CSE has to determine the most suitable candidate CSEs that contain the triple data matching with the ASD request. Since the data is distributed across CSEs in oneM2M network topology using multiple ontologies for triple data, there will be cases when a single SPARQL query may not return complete desired response from the candidate CSEs. Therefore, SPARQL query fragmentation is required, such that only sub-section of the query is sent to the candidate CSEs, which contain the relevant triple data. However, the hosting CSE has to make sure that even after the fragmentation of the SPARQL query, the logic defined in the original un-fragmented query remains intact, so that the expected response remains the same and can be aggregated.

Under this consideration, there can be different ways through which, a SPARQL query can be fragmented. One of these is fragmenting by using 'UNION' operator. In this case, the set of condition statements, grouped using 'UNION' operator, can be separated and redefined in a separate subsequent SPARQL query. Based on each subsequent query, different candidate CSEs can be selected from the SRT lookup. Table 6.2.3-1 shows the example of query fragmentation based on 'UNION' operator.

Table 6.2.3-1: SPARQL Query Fragmentation based on 'UNION' operator

Original SPARQL Query	Fragmented Subsequent SPARQL Queries

PREFIX base :
<http://www.onem2m.org/ontology/base-v1>
PREFIX xyz : <http://www.xyz.org/ontology/base-v1>
PREFIX abc : <http://www.abc.org/ontology/city-v1>
 	
SELECT ?device
WHERE {
{
?device a base:Device.
	PREFIX base :
	<http://www.onem2m.org/ontology/base-v1>
	PREFIX xyz : <http://www.xyz.org/ontology/base-v1>
	
	SELECT ?device
	WHERE {
	?device a base:Device.
	?device xyz:location city:Seongnam.
	}

```
    ?device xyz:location city:Seongnam.
   }
   UNION
   {
    ?device a xyz:Device.
    ?device xyz:location xyz:KETIHQ.
   }
   UNION
   {
    ?device a abc:Device.
    ?device abc:manufacturer ?manufacturer
    ?manufacturer abc:name company:Samsung
   }
}
```

```
PREFIX xyz : <http://www.xyz.org/ontology/base-v1>

SELECT ?device
WHERE {
   ?device a xyz:Device.
   ?device xyz:location xyz:KETIHQ.
}
```

```
PREFIX abc : <http://www.abc.org/ontology/base-v1>

SELECT ?device
WHERE {
   ?device a abc:Device.
   ?device abc:manufacturer ?manufacturer
   ?manufacturer abc:name company:Samsung
}
```

However, a SPARQL query can be defined in various different structures than the one defined above. Which may or may not ensure a valid fragmentation. Therefore, in addition to the considerations specified in Clause 6.2.2, following considerations should be adhered, while defining a fragmentable SPARQL query:

- The query should define common response to be returned by the CSEs, so that the result can be aggregated
- The WHERE Clause in the query should contain sub-clauses, which are connected together using 'UNION' operator.
- Each sub-clause should contain compulsory conditions to be matched in the triple data hosted by a CSE. These sub-clauses can be further fragmented if needed.
- It should be ensured that the logic of the query remains intact. That is, the logic of each subsequent query should not be changed if it defragmented back to the original query.
- It should be ensured that the requested the response to be returned by each subsequent query should have same final requested query variables, to ensure data aggregation.

The final consideration is demonstrated in Table 6.2.4-1, where the original as well as subsequent SPARQL queries have same query variable '?device'. This will ensure consistency in the response so that the returned list of devices is aggregated as if it was executed using the original unfragmented query. The advantage of this fragmentation is that it will simplify the complex queries and will lead to smaller query execution time in a target CSE.

6.2.4 Subsequent ASD Initiation

6.2.4.1 SPARQL query and target CSE

The subsequent ASD request can be defined and initiated after the Query fragmentation performed. In this case, a subsequent ASD request will be generated based on each subsequent SPARQL query. Each subsequent ASD request will have its own list of target CSEs where it will be sent.

Figure 6.2.4-1 shows the subsequent ASD request initiation based on the example in Table 6.2.4-1. Here, the outlined information in red, green, and blue, highlights the matched values as well as the target CSE for respective subsequent queries. Unlike the other subsequent queries, the query highlighted in green has three conditions, which are not completely matched with the corresponding RDF resource summary. In this case, a matching threshold can be considered to select the particular CSE for forwarding. If the number of conditions matched are equal or above the threshold, then the CSE should be considered. The threshold value is dependent on the hosting CSE's policy of target selection and may vary from on CSE to another. Here it can be observed that subsequent query is forwarded to only relevant CSEs. If instead, the original unfragmented query was used, then each CSE would execute the complete SPARQL query, searching for irrelevant triples, and may cause increase in the execution time.

Following the current specification, SPARQL queries will be carried in *semanticsFilter* condition of the ***Filter Criteria*** parameter. A CSE-ID for a subsequent ASD is used to build the ***To*** parameter value. If an ASD targets CSE2, for instance, the ***To*** parameter value will be "CSE2/CSEBaseName/asd".

In this example, the "CSEBaseName" is the resource instance name of <CSEBase> resource of CSE2. oneM2M provides a universal short name which is fixed in the spec as "-". So the example above can be re-written as "CSE2/-/asd". Also "ASD" represents the virtual resource, which will also be the fixed resource name in the spec, that handles ASD.

```
PREFIX xyz : <http://www.xyz.org/ontology/base-v1>

SELECT ?device
WHERE {
        ?device a xyz:Device.
        ?device xyz:location xyz:KETIHQ.
}
```

```
PREFIX abc : <http://www.abc.org/ontology/base-v1>

SELECT ?device
WHERE {
        ?device a abc:Device.
        ?device abc:manufacturer ?manufacturer
        ?manufacturer abc:name company:Samsung
}
```

```
PREFIX base : <http://www.onem2m.org/ontology/base-v1>
PREFIX xyz : <http://www.xyz.org/ontology/base-v1>

SELECT ?device
WHERE {
        ?device a base:Device.
        ?device xyz:location city:Seongnam.
}
```

#	CSE-ID	RDF Resource Summary
1	CSE2	http://www.xyz.org/ontology/base-v1#Device http://www.xyz.org/ontology/base-v1#Service http://www.xyz.org/ontology/base-v1#location
2	CSE3	http://www.abc.org/ontology/city-v1#Device http://www.abc.org/ontology/city-v1#Function http://www.abc.org/ontology/city-v1#manufacturer
3	CSE4	http://www.onem2m.org/ontology/base-v1#Device http://www.xyz.org/ontology/base-v1#location

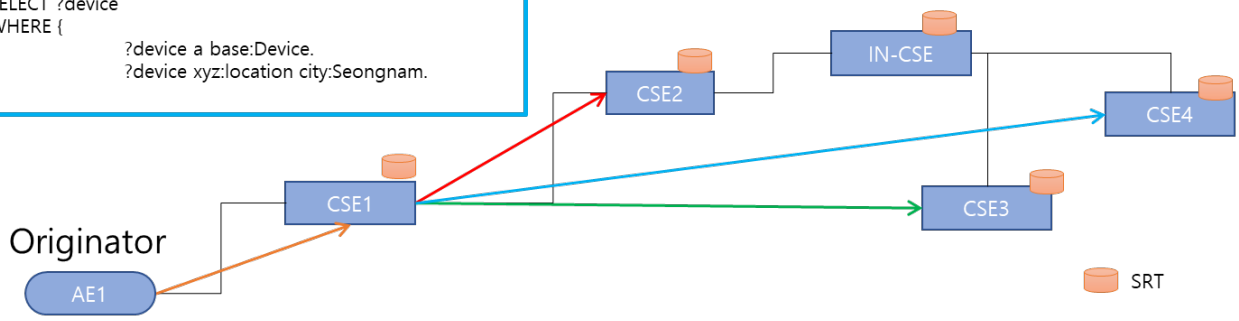

Figure 6.2.4.1-1: Subsequent ASD Initiation with matching summary

In a case there can be no match in a SRT for the ASD query. In this case, by default, the ASD handling CSE can fan-out the ASD to the list of CSEs in its SRT. However, if the Originator explicitly includes the *ASDMulticast* indicator as false, then there will be direct response back to the Originator with empty content.

6.2.4.2 Other parameters

In ASD request handling procedures, other than the ***Filter Criteria*** for SPARQL query and the ***To*** parameter for subsequent ASDs, there involves other request parameters as well. Those are defined in Clause 6.2.5 and those parameter settings from the original ASD will be referred to set the new value in the subsequent ASD.

6.2.5 ASD request parameter handling

6.2.5.1 Overview

Apart from the SPARQL query mentioned above, other request parameters can be included in a ASD request. Those are needed to indicate scope of CSEs, time limitation, result type, etc.

6.2.5.2 ASD handling indicator

This new optional parameter indicates the matching criteria of RDF resource summary in the SRT. When the ASD handling CSE selects target CSE(s) from its SRT, the query in the ASD request can be matched in ontology or type level. Also semantic inferencing can be performed to select target CSE(s). Default value is "type".

6.2.5.3 ASD target limit

This new optional parameter indicates the scope of targeted CSEs for advanced semantic discovery fan-out. This would avoid the discovery handling overhead for CSEs. There would be the system default value which is used when it is not provided in ASD requests.

The scope can be defined as the number of CSEs or hops from the first target CSE.

6.2.5.4 Subsequent ASD allowed

This new optional parameter indicates whether there could be subsequent ASD requests to other CSEs by the ASD handling CSE. Firstly, the Originator of ASD can choose whether this is allowed or not. If this is allowed so there can be several ASD fan-outs, the Originator basically says it is okay to wait for some time but expects a better result. Also, when an ASD handling CSE decides there are not enough target CSEs that it has chosen, it can allow the target CSEs to send out subsequent ASD request again. It may take other parameter values for consideration such as timestamps in the following sub-clauses.

6.2.5.5 ASD multicast allowed

This new optional parameter indicates when there is no match in a SRT, the ASD handling CSE fans-out the ASD request to all CSEs in its SRT. Default value is true.

6.2.5.6 Request expiration timestamp

The ***Request Expiration Timestamp*** request parameter defines the request validity time. When a request is received later than the timestamp, the request gets rejected. This existing parameter can be used to decide whether to perform the fan-out to other CSEs or execute the query locally for response.

6.2.5.7 Result expiration timestamp

The ***Result Expiration Timestamp*** request parameter defines the timestamp for the result validity by the Originator. When a response is received by the Originator after the timestamp, the result in the response gets ignored. This existing parameter can be used to set a timer to wait the fan-out semantic discovery responses. When the advanced semantic discovery request gets fanned-out, the original ***Result Expiration Timestamp*** value can be used to set the new ***Result Expiration Timestamp*** value, that will be earlier than the original one for aggregation, in the new requests.

6.2.5.8 Response type

The ***Response Type*** request parameter defines the communication mode between the Originator and the Hosting CSE. A complex discovery query would require a quiet long time to get the response back. When a non-blocking mode for this parameter is used, the Originator can get the response by its result retrieval (i.e. non-blocking synchronous mode) or by a notification (i.e. non-blocking asynchronous mode). This existing parameter can be used to by the Originator when the discovery is expected to take some time.

6.2.5.9 Event category

The ***Event Category*** request parameter defines the request handling priorities. The specification defines several category values and also other values can be defined for implementation [i.5]. This existing parameter can be used to set the priority among advanced semantic discovery requests.

6.2.5.10 Group Request Identifier

The ***Group Request Identifier*** request parameter is used to detect and avoid duplicated handling during group fan-outs. This parameter can be also used for the ASD fan-out to detect and avoid unwanted ASD fan-out loops.

6.2.6 Semantic Query Execution

Upon receiving the ASD request, the CSE will execute the SPARQL query in its local database. The CSE can execute the SPARQL query, which have stored the triple data and can interpret and execute query based on SPARQL 1.1 protocols. Selection of target CSE is performed using the SRT. Based on the SRT, if the hosting CSE is selected, it can execute the query in its local database as well as send the ASD or subsequent ASD to the other selected CSE using the SRT. The CSE will handle only the ASDs which are targeted towards it.

6.2.7 Result aggregation

The result aggregation involves the aggregation of SPARQL result and the corresponding oneM2M resources. The considerations specified in 6.2.4, simplifies the process of SPARQL result aggregation as it only involves merging of the results. However, the response can be aggregated either in multiple steps by different CSEs, or can be aggregated directly by the hosting CSE. In the former case aggregation depends upon different factors such as network topologies, SDAs, etc. For this case same example in figure 6.2.5-1 can be considered where the aggregation will be performed in two steps. In first step, the results of CSE3 and CSE4 can be aggregated either at the IN-CSE, or it can be aggregated at CSE2 along with the result of CSE2. Also, in this case the topology plays an important role as the CSEs involved in executing subsequent ASD queries lie on the path. In alternate case, the result will be aggregated directly at the hosting CSE.

```
PREFIX xyz : <http://www.xyz.org/ontology/base-v1>

SELECT ?device
WHERE {
        ?device a xyz:Device.
        ?device xyz:location xyz:KETIHQ.
}
```

Triples matched by subsequent query in CSE-2		
http://www.xyz.org/ontology/base-v1#smartwatch_be236	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.xyz.org/ontology/base-v1#Device
http://www.xyz.org/ontology/base-v1#smartwatch_be236	http://www.xyz.org/ontology/base-v1#location	xyz:KETIHQ

```
PREFIX abc : <http://www.abc.org/ontology/base-v1>

SELECT ?device
WHERE {
        ?device a abc:Device.
        ?device abc:manufacturer ?manufacturer
        ?manufacturer abc:name company:Samsung
}
```

Triples matched by subsequent query in CSE-3		
http://www.abc.org/ontology/city-v1#galaxyS21_4b85	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.abc.org/ontology/city-v1#Device
http://www.abc.org/ontology/city-v1#galaxyS21_4b85	http://www.abc.org/ontology/city-v1#manufacturer	http://www.abc.org/ontology/city-v1#SamsungSouthKorea
http://www.abc.org/ontology/city-v1#SamsungSouthKorea	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.abc.org/ontology/city-v1#Manufacturer
http://www.abc.org/ontology/city-v1#SamsungSouthKorea	http://www.abc.org/ontology/city-v1#name	company:Samsung

```
PREFIX base : <http://www.onem2m.org/ontology/base-v1>
PREFIX xyz : <http://www.xyz.org/ontology/base-v1>

SELECT ?device
WHERE {
        ?device a base:Device.
        ?device xyz:location city:Seongnam.
}
```

Triples matched by subsequent query in CSE-4		
http://www.onem2m.org/ontology/base-v1#thermometer_18gh0	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.onem2m.org/ontology/base-v1#Device
http://www.onem2m.org/ontology/base-v1#thermometer_18gh0	http://www.xyz.org/ontology/base-v1#location	city:Seongnam

Figure 6.2.7-1: SPARQL result example of subsequent query.

Figure 6.2.7-1 shows the example of the triples matched based on each subsequent query. Although multiple triples are matched based on different query parameters, the final query variable of each query is same, which in this case is '?device', containing the instance URIs of Devices, represented using three different ontologies. The result of each CSE is then returned in terms of this query variable, which are then aggregated at the hosting CSE1. Figure 6.2.7-2 shows the aggregation of these results of the above mentioned subsequent query execution. This is the simplified case involving only one query variable, where the aggregated result is generated by merging each result. However, there can be cases where different query parameters and processes such as grouping, adds to the complexity of aggregation process. Nonetheless, it can be handled by the CSE and the same process can be followed.

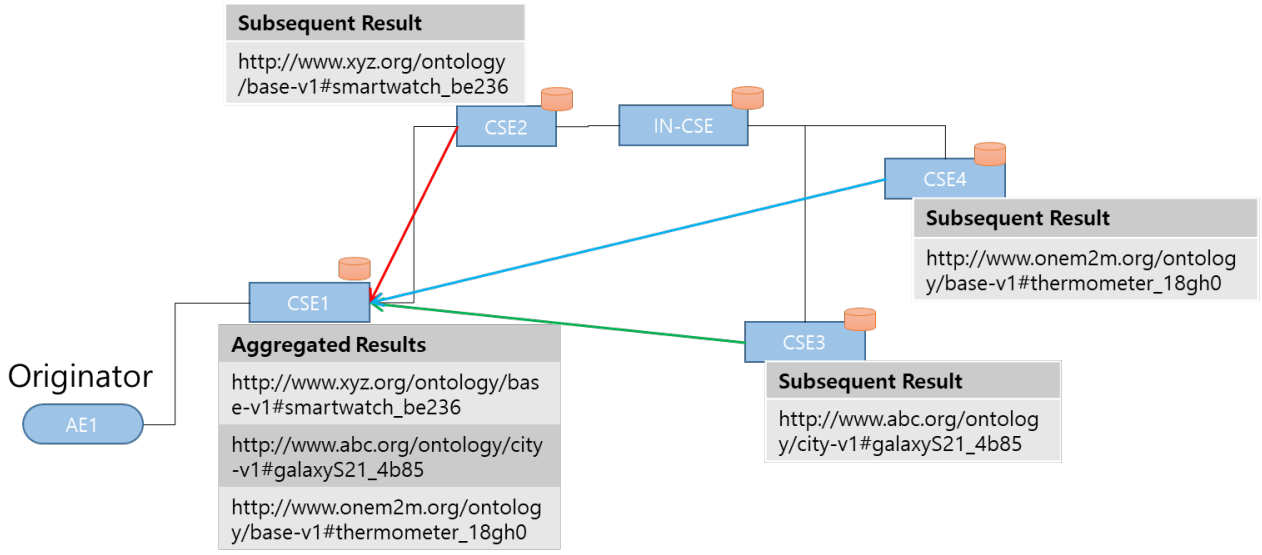

Figure 6.2.7-2: SPARQL result aggregation example.

6.3 Semantic discovery agreement

6.3.1 Relationships

Making a semantic discovery agreement between CSEs is to agree on the discovery fan-out scope and routing information access. A semantic discovery relationship in this context serves as defining a relation type for the fan-out scope and routing table access control. For interoperability, a relationship type needs to be specified in the specification, so different implementation can use the same type values for the same handling procedure.

During the semantic discovery routing, the relationships are used to limit the scope of discovery request forwarding or routing. For instance, a semantic discovery request from the CSE1 can be fanned-out to all the CSEs in the same M2M Service Provider domain.

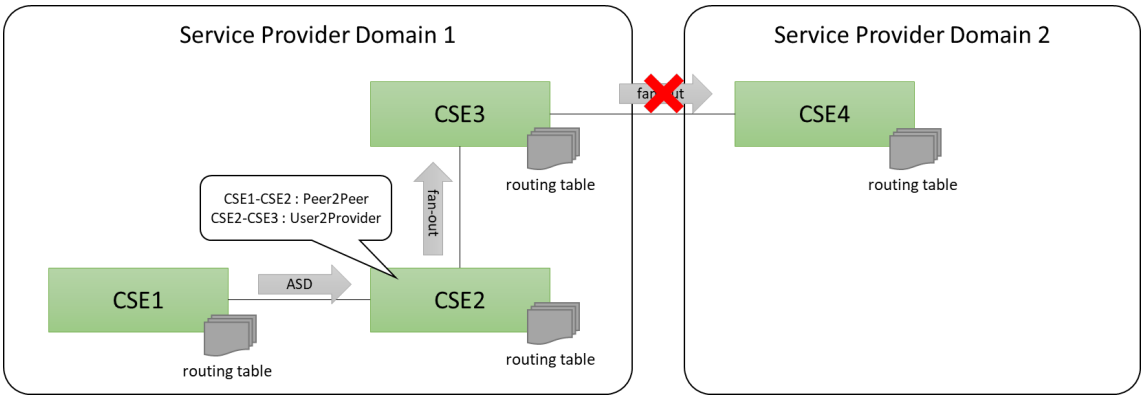

Figure 6.3.1-1: Example of semantic discovery relationship

6.3.2 Access Control

For routing record updates and propagations that are described in clause 6.3.4, a CSE needs to access (e.g. RETRIEVE for <subscription> resource creation) to other CSEs' resources. The relationships between CSEs can be used for access decision for the resources having semantic discovery routing information.

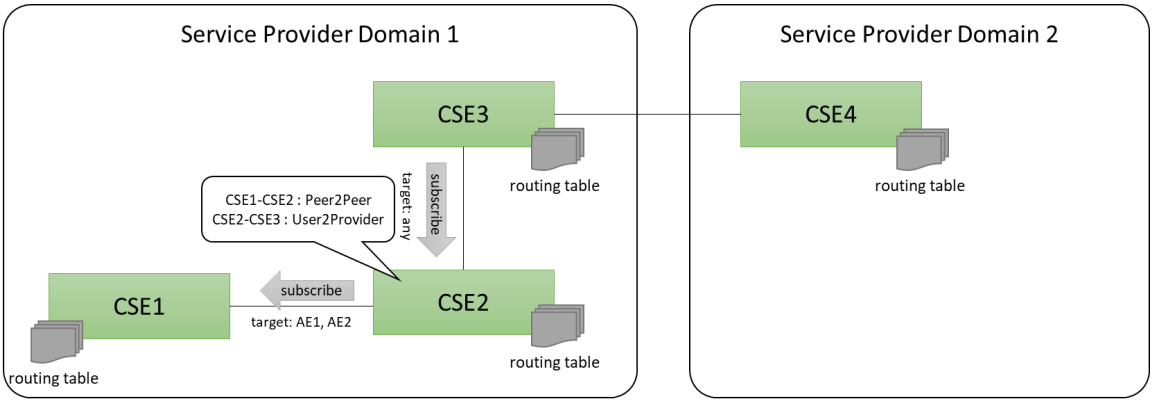

Figure 6.3.2-1: Example of semantic discovery access control

6.4 Semantic discovery routing recommendation

There has been a candidate solution for Semantic Recommendation System in Clause 7.2.7 [i.24]. It assumes a simple query involving a single type information, sensor type in the example, and suggested a P2P network based algorithm. However, the queries possibly supported by an ASD which includes SPARQL queries could be much more complicated than that. There can be different algorithms for many different aspects, so it seems to be more implementation specific so far. This is for further study to find standardizable solution on this topic.

7 Extensions to oneM2M specifications

7.1 Overview

Clause 7 proposes oneM2M specifications for the ASD features which have been illustrated in this document. This clause is structured to show how different aspects of the ASD solution gets incorporated into existing oneM2M standards. As other semantic features have been specified in the dedicated TS [i.25], while brief summary and interface definitions are defined in the architecture TS [i.26]. Following this principle, main feature specifications are corresponding to the Semantics Solutions TS [i.27]. Interface and protocol extensions pertaining to the specification documents [i.28][i.29].

Note that though this present document is the technical report, this clause uses normative verbs to propose oneM2M solution specifications for the oneM2M technical specifications.

7.2 Procedures to oneM2M TS-0034

7.2.1 Introduction of ASD

This clause describes the Advanced Semantic Discovery (ASD) which advances existing semantic discovery and filtering features in clause 7.4 [TS-34]. The differences between the semantic discovery and filtering are summarised in clause 7.5 [TS-34]. The ASD covers both mechanisms with advancements which are explained in the following clauses. Therefore the delimiter, whether a request is for semantic discovery or filtering, defined already in oneM2M specifications as the ***Semantic Query Indicator*** request parameter is also used in the ASD by the Originator to get different query results.

Similar to the existing mechanisms, the ASD also discovers and filters semantic descriptions in RDF triples with SPARQL queries. The targeted semantic data are contained in <*semanticDescriptor*> resources on a CSE as defined in the specification. Compared to the semantic discovery and filtering, which targets a priori known semantic resource instances (e.g. by a <*semanticDescriptor*> or <semanticFanOutPoint> resource), the ASD can target unknown semantic resource(s) which are distributed in more than one CSEs.

To request an ASD, the Originator set the request target (***To*** parameter) as <advancedSemanticDiscovery> virtual resource. The new virtual resource, which is different target resource from the semantic discovery and filtering, is defined in clause 7.3.

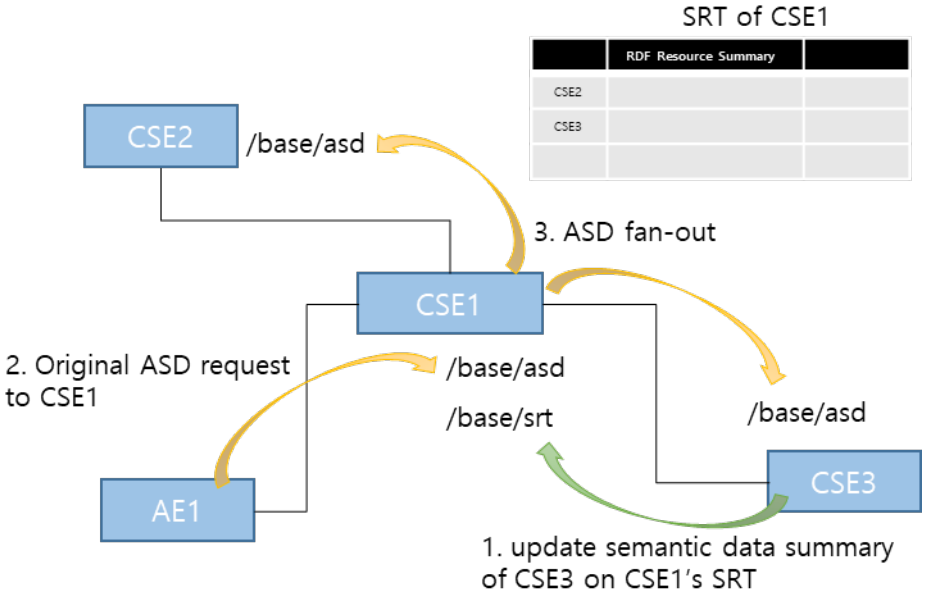

Figure 7.2. 1-1: ASD overall illustration

7.2.2 Semantic Routing Table

To execute an ASD on distributed CSEs, which are not known by the Originator, the Semantic Routing Table (SRT) is hosted and used by ASD handling CSEs. When a CSE receives an ASD request, the CSE looks for CSE(s) on its own Semantic Routing Table to further process the ASD by those CSEs. To select CSE(s), who get the ASD from the ASD handling CSE, the table keeps RDF resource summary provided by other CSEs.

The Table 7.2.2-1 defines the information elements of Semantic Routing Table. A new table record is created or updated logically, because the SRT is not represented as a oneM2M resource, either by Subscription/Notification mechanism or an Update request. When a CSE subscribes the other CSE's RDF resource summary (i.e. <*semanticRoutingTable*> virtual resource defined in clause 7.3), the CSE gets a notification per summary update with corresponding criteria setting. To handle the notifications, internal updates to the SRT, the *notificationURI* attribute of the subscription is set as the resource identifier of <*semanticRoutingTable*> virtual resource. In case of explicit Update requests, the ***Content*** parameter includes a record and targets the <*semanticRoutingTable*> virtual resource. Special handling on this virtual resource is specified in clause 7.3.

Table 7.2.2-1: Information on Semantic Routing Table record

Element	Description
CSE-ID	Identifier of the CSE which holds RDF triples that are corresponding to the RDF resource summary.
RDF Resource Summary	List of URIs, representing types of semantic data instances (i.e. RDF Class or a RDF Property [i.30]). Optionally this contains the number of instances per RDF type URI. E.g. "http://www.abc.org/ontology/city-v1#SmartDevice"
CSE Context Information	Optional. A set of context information on the CSE, other than RDF resource summary. See the table 7.2.2-2.

When there is a need to select a part of CSE(s) after RDF resource summary matching, CSE context information is used as the criteria to select a CSE over others. For instance there could be an ASD request for maximum 10 resource identifiers in the response.

Table 7.2.2-2: Types of CSE context information regarding ASD

Element	Description
Number of Hops	The number of hops between the SRT hosting CSE and the semantic data hosting CSE.
Semantic Discovery Agreement	Two CSEs agree on a relationship type which is used during the semantic routing. There is a priority among the types so the ASD handling CSE prefers a specific type over others. Types are defined by each M2M Service Provider. E.g. Customer-Provider, Peer-Peer

It is not the decision by a CSE itself on how to summarize semantic data and where to propagate the updates that are provided, but pertaining policies are given by AEs and the CSE summarizes and propagate new semantic data. The policies are configured on a <AE> resource (See clause 7.3).

Table 7.2.2-3: Policies for AEs regarding SRT

Type	Description
Summary Policy	This policy defines which semantic resources get summarized. Possible values are (other than "None", more than one can be selected): ● None ● Semantic resources under the <AE> resource ● Semantic resources created by the AE ● Semantic resources hold by the AE
Propagation Policy	This policy defines which CSEs can get RDF resource summary propagations. Possible values are (other than "None", more than one can be selected): ● None ● Registrar and Registry of the Hosting CSE ● CSEs that have specific semantic discovery agreements with the Hosting CSE (e.g. Provider)

7.2.3 ASD handling procedure

7.2.3.1 ASD request from Originator

The Originator sends an ASD request with the following parameters.

- ***To***: The <*advancedSemanticDiscovery*> resource identifier on a CSE. (i.e. "{CSE-ID}/{CSEBase-name}/asd")
  Note that "asd" is the short name of <*advancedSemanticDiscovery*> virtual resource (see clause 7.3)
- ***Filter Criteria***: the *semanticsFilter* condition contains a SPARQL query and the *limit* condition to indicate the max number of resource IDs in the response
- ***Semantic Query Indicator***: "TRUE" for SPARQL query results, or "FALSE" for corresponding oneM2M resource identifiers in the response
- ***ASD Handling Indicator***: When the Originator prefers as many results as possible while waiting for some time, "ontology URI" can be set. If better performance with shorter time limit is given "type URI" can be chosen. When the Originator has not enough information while composing the SPARQL query "semantic inferencing" can be chosen, which provides inferencing to select ASD executing target CSE(s).
- ***ASD Target Limit***: limits to the number of CSEs or the number of hops from the Hosting CSEs that executes the query in the ASD
- ***ASD Subsequent Fan-out Allowed***: "TRUE" if the Originator allows the ASD fan-out on CSEs after the initial fan-out on the Hosting CSE
- ***ASD Multicast Allowed***: "TRUE" if the Hosting CSE is allowed to fan-out the ASD to the CSEs in the Hosting CSE's SRT when there is no match in the SRT
- Other timestamp parameters (i.e. ***Request Expiration Timestamp, Result Expiration Timestamp***) and request handling related request parameters (i.e. ***Response Type, Event Category***)

7.2.3.2 SPARQL Query Fragmentation

Before selecting target CSEs to handle the ASD, the SPARQL query fragmentation is performed by the Hosting CSE in order to avoid unnecessary query fan-out to multiple CSEs. The Hosting CSE fragments the original query in the ASD request into more than one independent subsequent queries which are then sent to different target CSEs based on the respective lookup result of each query.

One of the considered approaches for SPARQL query fragmentation is by UNION logical operator defined in the query. UNION operator is used as a fragmentation point of the query, whereas the other statements defined in conjunction using the AND logical operator remain together in the subsequent query.

Certain essential statements in the query such as SELECT statement, PREFIX statement which form the main building block of SPARQL query are duplicated for each subsequent query. However, it is ensured that the response type and format remain the same throughout the subsequent queries of the particular ASD. Figure 7.2.2.2-1 shows the general structure of query fragmentation using UNION operator as fragmentation point.

After performing the fragmentation, the process of SRT lookup defined in clause 7.2.2.3 is then executed for each subsequent SPARQL query independently.

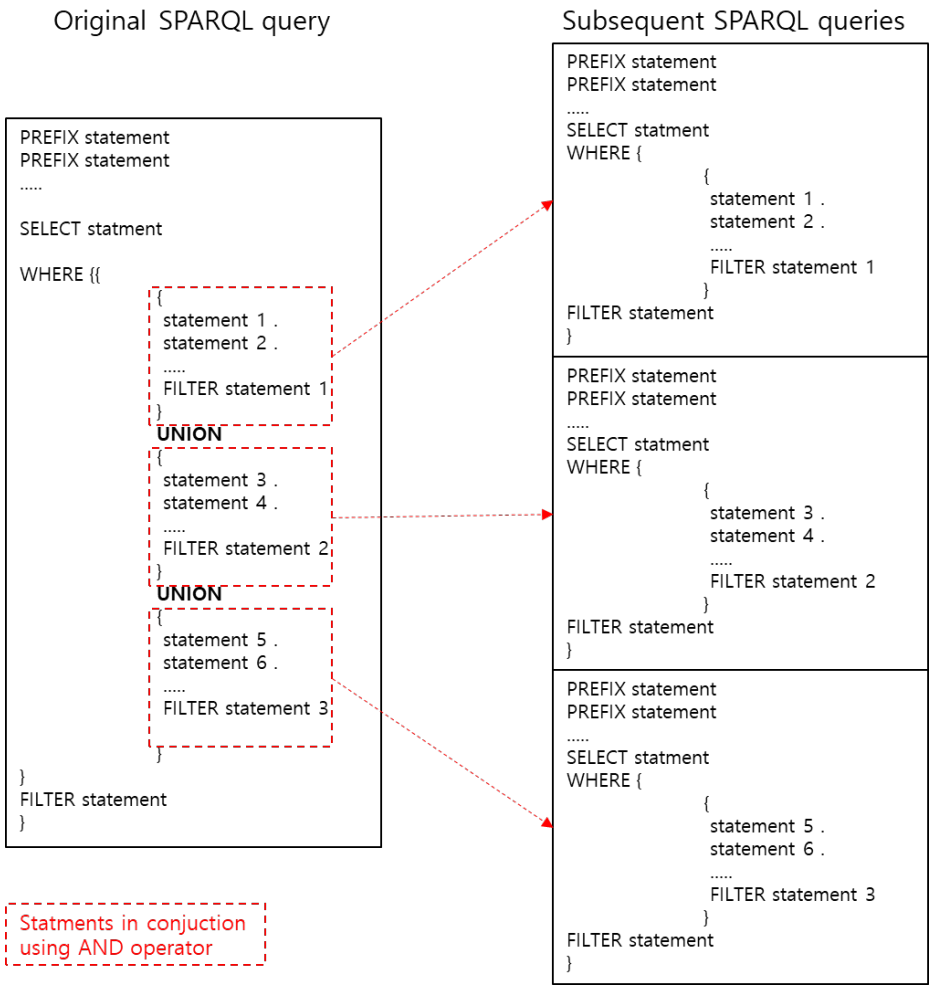

Figure 7.2.3.2-1: SPARQL Query fragmentation using UNION Operator

7.2.3.3 ASD target selection based on matching criteria

After the query fragmentation of the ASD, the Hosting CSE shall select CSEs from the list in the SRT to fan-out the ASD.

From ASD query, the Hosting CSE shall check the ***ASD Handling Indicator*** parameter, which specifies the matching criteria to be considered while selecting the target CSEs. The matching criteria specifies the information to be considered in the summary of the triple data, for selecting the target CSEs. Depending on the ***ASD Handling Indicator*** parameter value, following different target selection scheme shall be performed:

Case 1: The ***ASD Handling Indicator*** parameter specifies "ontology URI" to be considered for selecting the target CSE in the SRT. In this case, the Hosting CSE shall extract the ontology URIs, included in the SPARQL query, and performs the lookup of the summary in the SRT for these ontologies. The ontology URI are extracted from either the prefixes specified in the SPARQL query or from the entity URIs specified in the query conditions in any part of SPARQL query. Upon finding the matched ontology URIs, the corresponding CSEs are selected as target for the ASD to be fanned-out.

Case 2: The ***ASD Handling Indicator*** parameter specifies "type URI" in the summary to be considered for selecting the target CSE in the SRT. In this case, the Hosting CSE shall extract the relevant attributes from the SPARQL query given in the ASD request, such as ontology URI, RDF Class URI, RDF Property URI, etc., and then performs the loop of the summary in the SRT for these matching attributes. The RDF Class or Property URIs are extracted from the entity URIs specified in the query conditions in any part of SPARQL query.

Case 3: In addition to the detailed summary, the ***ASD Handling Indicator*** parameter specifies "semantic inferencing" to be applied in order to select the additional target CSEs in the SRT, which are filtered out in the above-mentioned selection criteria. In this case, the Hosting CSE shall perform the tasks defined in case 2, that is, extracting the attributes to be matched with the summary in the SRT, such as ontology URI, RDF Class URI, RDF Property URI, etc. In addition, those entries in the SRT are listed, whose ontology URIs are matched, but no RDF Class URI or RDF Property

URI is matched. Then, the Hosting CSE shall access (e.g. get the ontology from the web) those matched ontology to perform inferencing, from which additional information (e.g. assertions such as rdfs:subClassOf, rdfs:subPropertyOf, etc.) is extracted, to identify whether the listed CSEs are valid targets for the given SPARQL query. From the list, the filtered CSEs are then added to the complete list of target CSEs.

As described in clause 6.2.4.1, for each case (i.e. case 2 and 3) a threshold can be defined to have the least number of possible matches at the basic level of ASD target selection process, which is subject to implementation methodologies.

7.2.3.4 ASD fan-out

When there exists target CSE(s), from the query matching in clause 7.2.2.3, to further process the ASD, the Hosting CSE shall fan-out the ASD to those CSE. The request parameters shall be set as follows:

- ***To***: The <*advancedSemanticDiscovery*> resource identifier on the target CSE. (i.e. "{Target-CSE-ID}/{CSEBase-name}/asd")
- ***From***: CSE-ID of the Hosting CSE that is handling the ASD
- ***Filter Criteria***: the *semanticsFilter* condition contains a SPARQL query, this may be the fragmented query (see Clause 7.2.2.2). the *limit* condition can be updated from the original ASD request.
- ***Semantic Query Indicator***: the same value as the original ASD
- ***ASD Handling Indicator***: the same value as the original ASD
- ***ASD Target Limit***: updated value of the number of CSEs or hops
- ***ASD Subsequent Fan-out Allowed***: when there is enough time to further fan-out the ASD this can be set as "TRUE"
- ***ASD Multicast Allowed***: the same value as the original ASD
- Other timestamp parameters (i.e. ***Request Expiration Timestamp, Result Expiration Timestamp***) are updated and request handling related request parameters (i.e. ***Response Type, Event Category***).

When the Hosting CSE receives the response(s) for the ASD, the Hosting CSE shall aggregate results and send the response back to the Originator who requested the original ASD. When the ASD requests SPARQL query results, not the oneM2M resource identifiers, in the response using ***Semantic Query Indicator*** parameter, it could not always possible to aggregate results. This is for further study.

7.3 Interface extensions to oneM2M TS-0001

7.3.1 New resource type *advancedSemanticDiscovery*

7.3.1.1 Resource type definition

The <*advancedSemanticDiscovery*> resource is a virtual resource because it does not have a resource representation.

When a retrieve request is sent to the <*advancedSemanticDiscovery*> resource, it is considered to perform the Advanced Semantic Discovery (Clause 7.2.3). The Hosting CSE shall select target CSE(s), including the Hosting CSE itself, from the local Semantic Routing Table with the SPARQL query contained in the request. During the target selection, the query fragmentation may be performed to efficiently perform the ASD on distributed CSEs. When the targets are selected, the Hosting CSE shall fan-out the ASD and send back the aggregated results to the Originator.

The short name for this virtual resource shall be 'asd'.

7.3.1.2 Procedures

The <*advancedSemanticDiscovery*> resource only defines the retrieve operation, other operations are not allowed.

Table 7.3.1.2-1: *<advancedSemanticDiscovery>* RETRIEVE

***<advancedSemanticDiscovery>* RETRIEVE**	
Information in Request message	***Filter Criteria:*** the *semanticsFilter* condition contains a SPARQL query
Processing at Originator before sending Request	Same as the general procedure.
Processing at Receiver	The Hosting CSE may fragment the SPARQL query in the request. Then the Hosting CSE shall select CSE(s) in the SRT by matching the query or fragmented queries with RDF summaries of the CSEs. If there is one or more match(es), the Hosting CSE shall fan-out the ASD to those CSEs and the target resource shall be *<advancedSemanticDiscovery>* resource on those CSEs. The SPARQL query in the ***Filter Criteria*** of the fanned-out request may be the fragmented queries, otherwise the same query from the Originator's request. If there is no match, but the ***ASD Multicast Allowed*** parameter is "TRUE", then the Hosting CSE shall fan-out the request to the CSE(s) in the local SRT. If there is no match, and the ***ASD Multicast Allowed*** parameter is "FALSE", then the Hosting CSE shall send successful response with empty ***Content*** parameter.
Information in Response message	Same as the general procedure, except the following. ● If there was the fan-out by the Hosting CSE, the Hosting CSE shall aggregate the responses and send the aggregation response to the Originator. ● Depending on the ***Semantic Query Indicator*** parameter from the request, the ***Content*** shall contain the list of resource identifiers or the SPARQL query results.
Processing at Originator after receiving Response	None
Exceptions	None

7.3.2 New resource type *semanticRoutingTable*

7.3.2.1 Resource type definition

The <*semanticRoutingTable*> resource is a virtual resource because it does not have a resource representation.

When an update request is sent to the <*semanticRoutingTable*> resource, it is considered to propagate or update the RDF resource summary on the SRT of the Hosting CSE. Therefore this update request can be used by the CSE that has semantic data and propagates the summary of that data to other CSEs per *advancedSemanticDiscoveryPolicy* on a <AE> resource.

On the other hand, a CSE can subscribe the <*semanticRoutingTable*> resource of another CSE, which has semantic data. When the subscribed-to resource it this virtual resource and the *notificationEventType* of the <*subscription*> resource, the Hosting CSE shall notify the subscription-made CSE per summary update event.

The short name for this virtual resource shall be 'srt'.

7.3.2.2 Procedures

The <*semanticRoutingTable*> resource only defines the update and notify operation, other operations are not allowed.

The update request is used to update the SRT of the Hosting CSE. The other CSE can propagate updated RDF resource summary or an Registry AE can provide RDF resource summary updates to its Registrar CSE. The routing record only pertaining to the Originator CSE gets updated.

Table 7.3.2.2-1: <*semanticRoutingTable*> UPDATE

<*semanticRoutingTable*> UPDATE	
Information in Request message	***Content:*** a Semantic Routing Table record of the Originator (See clause 7.2.2 for information structure of the record)
Processing at Originator before sending Request	Same as the general procedure.
Processing at Receiver	The Hosting CSE shall update the RDF Resource Summary and CSE Context Information as contained in the request ***Content*** parameter.
Information in Response message	Same as the general procedure.
Processing at Originator after receiving Response	None
Exceptions	None

The notify request is used to send the updated RDF resource summary to another CSE.

Table 7.3.2.2-2: <*semanticRoutingTable*> NOTIFY

<*semanticRoutingTable*> NOTIFY	
Information in Request message	***To***: Identifier of the <*semanticRoutingTable*> resource ***Content***: an updated RDF resource summary of the Originator
Processing at Originator before sending Request	Same as the general procedure.
Processing at Receiver	The Hosting CSE shall update the RDF Resource Summary as contained in the request ***Content*** parameter
Information in Response message	Same as the general procedure.
Processing at Originator after receiving Response	None
Exceptions	None

7.3.3 Extensions to resource type *AE*

This clause defines the additional attribute to the *AE* resource type for the ASD handling.

Table 7.3.3-1: Attributes of <*AE*> resource

Attributes of <*AE*>	Multiplicity	RW/ RO/ WO	Description	<*AEAnnc*> Attributes
semanticRoutingTablePolicy	0..1	RW	This policy is used by the Hosting CSE to perform RDF resource summary for the semantic resource of the AE and the summary propagation to other CSEs. Possible values for the summary policy are (other than "None", more than one can be selected): ● None ● Semantic resources under the <AE> resource ● Semantic resources created by the AE ● Semantic resources hold by the AE Possible values of the propagation policy are (other than "None", more than one can be selected): ● None ● Registrar and Registry of the Hosting CSE ● CSEs that have specific semantic discovery agreements with the Hosting CSE (e.g. Provider)	NA

7.3.4 Extensions to resource type *Subscription*

This clause defines the additional attribute to the *subscription* resource type for the ASD handling.

Table 7.3.4-1: *eventNotificationCriteria* conditions

Condition tag	Multiplicity	Matching condition
notificationEventType	0..6	H. When the RDF resource summary of the Hosting CSE gets updated, a notification is generated. This event type can be used with the *notificationContentType* attribute. When the attribute value is "modified attribute", the Hosting CSE shall send the updated RDF resource summary of the Hosting CSE, while "all attributes" delivers the entire summary of the Hosting CSE. (Note that oneM2M TS-0001 already defines event types from A to G already and this proposes the new type)

7.3.5 New request parameters

7.3.5.1 ASD handling indicator

This new optional parameter indicates the matching criteria of RDF resource summary. By the Originator's indication, the query in the ASD request can be matched in ontology or type level with the RDF resource summary in the SRT to select target CSE(s). Also semantic inferencing on the summary can be performed to select target(s). Default value is "Type".

7.3.5.2 ASD target limit

This new optional parameter indicates the scope of targeted CSEs for advanced semantic discovery fan-out. The scope can be defined as the number of CSEs or hops from the first target CSE.

7.3.5.3 ASD subsequent fan-out allowed

This new optional parameter indicates whether there could be subsequent ASD request fan-outs to other CSEs by the ASD handling CSE.

7.3.5.4 ASD multicast allowed

This new optional parameter indicates when there is no match in a SRT, the ASD handling CSE fans-out the ASD request to all CSEs in its SRT. Default value is true.

7.4 Protocol extensions to oneM2M TS-0004

7.4.1 Data types

7.4.1.1 Primitive parameter data types

The data types of request primitive parameters are specified in this clause.

Table 7.4.1.1-1: Data Types for Request primitive parameters

Primitive Parameter	Data Type	Multiplicity	Default Handling	Note
ASD Handling Indicator	m2m:asdMatchingCriteria	0..1	"Type" URI matching	
ASD Target Limit	xs:positiveInteger	0..1	unlimited	
ASD subsequent fan-out allowed	xs:boolean	0..1	true	
ASD multicast allowed	xs:boolean	0..1	true	

7.4.1.2 Resource attribute data types

This clause defines additional attribute definition of the AE resource type.

Table 7.4.1.2-1: Resource Specific Attributes of <AE> resource

Attribute Name	Request Optionality		Data Type	Default Value and Constraints
	Create	Update		
semanticRoutingTablePolicy	O	O	m2m:semanticRoutingTablePolicy	No default

7.4.1.3 Complex data types

This clause defines the complex data types to support the ASD.

The table 7.4.1.3-1 defines the m2m:semanticRoutingTablePolicy for the *semanticRoutingTablePolicy* attribute of the *AE* resource type.

Table 7.4.1.3-1: Type Definition of m2m:semanticRoutingTablePolicy

Element Path	Element Data Type	Multiplicity	Note
summaryPolicy	m2m:srtSummaryPolicy	1	
propagationPolicy	m2m:srtPropagationPolicy	1	

The table 7.4.1.3-2 extends the existing m2m:notification data type to send the notification event when there is an update to a record in a SRT.

Table 7.4.1.3-2: Type Definition of m2m:notification

Element Path	Element Data Type	Multiplicity	Note
notificationEvent/srtRecord	m2m:srtRecord	0..1	

The table 7.4.1.3-3 defines the m2m:srtRecord data type which is used to in Update or Notify request targeting <*semanticRoutingTable*> resource.

Table 7.4.1.3-3: Type Definition of m2m:srtRecord

Element Path	Element Data Type	Multiplicity	Note
CSE-ID	m2m:ID	1	
RDF resource summary	m2m:rdfResourceSummary	1	
CSE context information	m2m:cseContextInformation	0..1	

The table 7.4.1.3-4 defines the m2m:rdfResourceSummary data type.

Table 7.4.1.3-4: Type Definition of m2m:rdfResourceSummary

Element Path	Element Data Type	Multiplicity	Note
URIs	m2m:listOfURIs	1	
cardinality	xs:positiveInteger	0..1	

The table 7.4.1.3-5 defines the m2m:cseContextInformation data type.

Table 7.4.1.3-5: Type Definition of m2m:cseContextInformation

Element Path	Element Data Type	Multiplicity	Note
number of hops	xs:nonNegativeInteger	0..1	
semantic discovery agreement	Note 1	0..1	

Note 1: Enumeration values are defined by a M2M Service Provider.

7.4.1.4 Enumeration data types

This clause defines new enumeration types and new values to support the ASD.

The table 7.4.1.4-1 defines the new enumeration value to the notificationEventType condition of the eventNotificationCriteria attribute of subscription resource type.

Table 7.4.1.4-1: Interpretation of notificationEventType

Value	Interpretation	Note
8	Update of RDF resource summary of the Hosting CSE	Only applicable when the subscribed-to resource is <*semanticRoutingTable*> resource

The table 7.4.1.4-2 and 7.4.1.4-3 define the new enumeration types m2m:srtSummaryPolicy and m2m:srtPropagationPolicy respectively for the *semanticRoutingTablePolicy* attribute of the *AE* resource type.

Table 7.4.1.4-2: Interpretation of srtSummaryPolicy

Value	Interpretation	Note
1	None	
2	Semantic resources under the <AE> resource	
3	Semantic resources created by the AE	
4	Semantic resources hold by the AE	

Table 7.4.1.4-3: Interpretation of srtPropagationPolicy

Value	Interpretation	Note
1	None	
2	Registrar and Registry of the Hosting CSE	

Note: Other values can be defined by the M2M Service Provider (e.g. Provider CSEs, Peer CSEs)

The table 7.4.1.4-4 defines the new enumeration types m2m:asdMatchingCriteria for the ***ASD Handling Indicator*** *parameter*.

Table 7.4.1.4-4: Interpretation of asdMatchingCriteria

Value	Interpretation	Note
1	Ontology	
2	Type	Default value
3	Semantic Inferencing	

7.4.2 Error handlings

There is no new oneM2M Response Status Code (RSC) definitions are needed. However the RCSs in the table 7.4.2-1 shall be used to handle error cases in the ASD.

Table 7.4.2-1: RSCs for the ASD

RSC Value	RSC Description	Error cases
4005	OPERATION_NOT_ALLOWED	When a request, other than Retrieve, is sent to the <*advancedSemanticDiscovery*> resource, the Hosting CSE shall return the OPERATION_NOT_ALLOWED error. When a request, other than Update or Notify, is sent to the <*semanticRoutingTable*> resource, the Hosting CSE shall return the OPERATION_NOT_ALLOWED error.
4118	ONTOLOGY_NOT_AVAILABLE	When the ontology retrieval for semantic inferencing (See clause 7.2.2.3) is not available, the Hosting CSE shall return the ONTOLOGY_NOT_AVAILABLE.
5001	NOT_IMPLEMENTED	When the Hosting CSE does not implement <*advancedSemanticDiscovery*> or <*semanticRoutingTable*> resource but receives a Retrieve or Update/Notify request, respectively, the CSE shall return NOT_IMPLEMENTED error.
5232	REASONING_PROCESSING_FAILED	When the semantic reasoning for semantic inferencing (See clause 7.2.2.3) is not available, the Hosting CSE shall return the REASONING_PROCESSING_FAILED.

8 Conclusions

The Advanced Semantic Discovery solution for oneM2M system has been developed in this present document as the continuation work to [i.31][i.32]. The solutions described in clause 6 and specified in clause 7 enables the short comings of existing semantic discovery and query with the semantic routing concept. Therefore, now, an AE can discover or query unknown CSEs by reaching out just one CSE who can handle the ASD.

Similar to the ontology choice, which is open for implementation, for semantic descriptions in CSEs, RDF resource summary is up to implementation choice. However, the ASD solution is still interoperable since the summary information is still uniquely identifiable so the ASD request is able to reach CSEs who eventually has matching semantic data for the ASD queries.

Also, the ASD provides different options for AE to provide the RDF resource summary pertaining to the resources belong to itself. When it requests ASDs, there are other options to be enforced to ASD handling CSEs (e.g. ASD handling indicator, ASD target limit).

Discovery is a basic feature for IoT system deployments. As the discovery functionality gets richer like the ASD, there should be bigger leverages to take for IoT service implementations.

Annex A:
Change History

Date	Version	Information about changes
2021.03.21	v0.3.0	Clause 5 and 6 updates including ASD multicast and subsequent fan-out mechanisms.
2021.04.29	v0.4.0	Clause 7 added and updates on existing texts/diagrams to the member's feedback.
2021.04.30	v0.5.0	Clause 4 added.
2021.05.12	v0.6.0	Provides further description on SRT and missing description on routing recommendation.
2021.05.18	v0.7.0	Terminology and reference clean-up, missing data types and object definitions, error handlings and others towards the conclusion.
2021.05.21	v0.8.0	Editorial updates to provide the final draft
2021.05.21	v0.8.1	Further modifications to provide the final draft for approval at the TC SmartM2M#58
2021.06.01	V1.1.1	Technical Officer review for EditHelp publication pre-processing

History

Document history		
V0.1.1	December 2020	Clean-up done by ***editHelp!*** E-mail: mailto:edithelp@etsi.org
V1.1.1	June 20201	Publication